YAYASAN PRIMA AGUS TEKNIK

Dr. Budi Raharjo, S.Kom, M.Kom.,MM.

KOMPUTASI HIJAU

(Green Computing)

Komputasi hijau (Green Computing)

Penulis :
Dr. Budi Raharjo, S.Kom., M.Kom., MM.

ISBN : 9 786238 120130

Editor :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Penyunting :
Dr. Joseph Teguh Santoso, M.Kom.

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom

Penebit :
Yayasan Prima Agus Teknik

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : info@stekom.ac.id

KATA PENGANTAR

Puji Syukur penulis panjatkan atas selesainya buku yang berjudul ***"Komputasi hijau (Green Computing)"***. Selama dekade terakhir kita telah menyaksikan peningkatan minat penyedia layanan komputasi dalam merancang model dan metodologi cerdas dan meningkatkan infrastruktur yang ada ke sistem komputasi berkinerja tinggi yang dapat memenuhi permintaan aplikasi baru yang semakin kuat. Secara paralel, hampir bersamaan, pabrikan komputasi telah mengkonsolidasikan dan berpindah dari server yang berdiri sendiri yang dipasang di rak. Sistem komputasi terdistribusi skala besar dengan parameter tinggi dapat terdiri dari sejumlah besar berbagai komponen (komputer, database, dll) dan harus menyediakan berbagai layanan, tidak terbatas hanya pada platform komputasi kinerja tinggi.

Pengguna yang terdistribusi secara geografis mungkin memiliki akses terbatas ke layanan dan sumber daya sistem dan persyaratan yang berbeda, seringkali bertentangan. Selain itu, juga informasi dan data yang diproses dalam lingkungan yang dinamis tersebut mungkin tidak lengkap, tidak tepat, terpisah-pisah, dan kelebihan beban. Semua masalah yang disebutkan di atas memerlukan pengembangan teknik manajemen sumber daya yang cerdas. Di sisi lain, sistem manajemen multi-level yang kompleks dapat secara langsung menyebabkan peningkatan penggunaan listrik dalam sistem tersebut. Pemanfaatan energi yang optimal telah mencapai titik di mana banyak manajer teknologi informasi (TI) dan eksekutif perusahaan bersatu padu untuk mengidentifikasi solusi terukur yang dapat mengurangi konsumsi listrik (sehingga total biaya operasi diminimalkan) dari masing-masing besar- skala sistem komputasi dan secara bersamaan meningkatkan atau mempertahankan throughput sistem saat ini.

Dua masalah mendasar yang harus diatasi saat mempertimbangkan solusi holistik yang akan menghasilkan alokasi sumber daya hemat energi dalam komputasi performa tinggi, yaitu (a) total biaya operasi (TCO) dan (b) pembuangan panas. Namun, kedua masalah ini dapat diselesaikan dengan alokasi (ulang) dan pengelolaan sumber daya yang hemat energi. Jika sama sekali tidak ada yang dilakukan, maka konsumsi daya per sistem akan meningkat yang pada gilirannya akan meningkatkan biaya listrik dan pemeliharaan. Sebagian besar listrik yang dikonsumsi oleh prosesor berperforma tinggi diubah menjadi panas, yang dapat berdampak parah pada kinerja keseluruhan sistem yang dikemas rapat.

Buku ini dengan ini menyajikan ide-ide kunci dalam analisis, implementasi, dan evaluasi teknik cerdas generasi berikutnya untuk perumusan dan solusi komputasi hijau yang kompleks dan masalah TI hijau secara umum. Solusi berbasis metaheuristik evolusioner dan umum yang cerdas untuk pengoptimalan energi dalam pemrosesan data, penjadwalan, alokasi sumber daya, dan komunikasi dalam grid modern, komputasi jaringan dan komputasi disajikan bersama dengan beberapa teknologi konvensional penting untuk membahas topik hangat dari dasar teori konsep "komputasi hijau" dan untuk menggambarkan arsitektur dasar sistem.

Buku ini terdapat delapan babnya yang disusun] menjadi tiga bagian utama: bagian pertama, di mana kita berada?: Masalah penjadwalan hemat energi, pemrosesan data, komunikasi dan alokasi sumber daya dalam sistem komputasi terdistribusi skala besar telah dipelajari secara intensif selama beberapa tahun terakhir. Namun, masih belum banyak contoh penerapan metodologi berbasis evolusi sadar energi yang dapat ditemukan dalam literatur pada Bab 1 memberikan survei komprehensif tentang *state-of-the-art* dalam komputasi hijau berbasis evolusioner dalam infrastruktur Fisik Cyber Terdistribusi (DCP). Serta menentukan masalah optimisasi global yang terkait dengan penggunaan energi kumulatif. Efektivitas teknologi yang dipilih diilustrasikan dalam studi kasus di masing-masing lingkungan yang dipertimbangkan.

Bagian ke 2 buku ini akan menjelaskan tentang kesadaran energi dalam sistem komputasi kinerja tinggi: Sistem Komputasi Kinerja Tinggi (HPC) modern dirancang sebagai platform multi-level untuk menyediakan berbagai layanan bagi pengguna. Manajemen catu daya dan konsumsi energi yang efektif dalam infrastruktur grid, cluster, dan cloud saat ini sangat bergantung pada struktur semua perangkat fisik, middleware, dan lapisan layanan pengguna virtual dalam arsitektur sistem. Bab 2 penulis merumuskan contoh masalah sesuai dengan persyaratan *Quality of Service* (QoS) dari pengguna dan penyedia layanan, melalui konsolidasi sumber daya dan teknik migrasi, atau dengan menyesuaikan keadaan node fisik. Dalam Bab 3 akan menganalisis model formal catu daya optimal untuk server dalam skala makro dan mikro di lingkungan HPC. Masalah peralatan sistem jaringan dengan perangkat komputasi berdaya rendah dengan modul catu daya modular dibahas dalam Bab 4. Penulis merumuskan masalah penjadwalan dalam jaringan komputasi sebagai tugas optimisasi global multi-tujuan dengan konsumsi energi dan mengamankan alokasi sumber daya sebagai kriteria penjadwalan utama. Akhir bagian ke 2 ditutup di Bab 5 yang mendefinisikan model generik dari teknik harga bayangan sederhana untuk masalah penjadwalan kendala tenggat waktu tugas.

Bagian terakhir dalam buku ini mencakup Bab 6 hingga Bab 8 yang akan menerangkan tentang Manajemen Sadar Energi di Banyak Sistem Inti dengan Nirkabel Komunikasi: Bab 6 membahas tantangan desain khusus dalam memantau suhu dalam sistem multi-inti berskala besar. Teknik komunikasi nirkabel berenergi rendah dalam sistem banyak-inti akan dijabarkan pada Bab 7. Bagian akhir sekaligus bab terakhir buku ini Bab 8 akan menerangkan tentang solusi evolusioner untuk dua masalah fundamental kompresi data dan lokalisasi node di Jaringan Sensor Nirkabel dengan server multi-core. Akhir kata semoga buku ini berguna bagi para pembaca.

Semarang, Januari 2023

Penulis

Dr. Budi Raharjo, S.Kom, M.Kom. M.M.

DAFTAR ISI

BAB 1
KOMPUTASI HIJAU EVOLUSIONER UNTUK SISTEM CYBER TERDISTRIBUSI

Sistem Fisik Cyber Terdistribusi (DCPS) adalah jaringan sistem komputasi yang memanfaatkan informasi dari lingkungan fisiknya untuk menyediakan layanan penting seperti kesehatan cerdas, komputasi awan hemat energi, dan jaringan cerdas. Memastikan operasi ramah lingkungan mereka, yang meliputi efisiensi energi, keamanan termal, dan operasi tanpa gangguan jangka panjang meningkatkan skalabilitas dan keberlanjutan infrastruktur ini. Mencapai tujuan ini seringkali membutuhkan peneliti untuk memanfaatkan pemahaman tentang interaksi antara peralatan komputasi dan lingkungan fisiknya. Memodelkan interaksi ini dapat menjadi tantangan komputasional dengan sumber daya yang tersedia dan kebutuhan pengoperasian sistem tersebut. Untuk mengatasi kesulitan komputasi ini, para peneliti telah menggunakan Algoritma Evolusioner (EA), yang menggunakan pencarian acak untuk menemukan solusi yang mendekati optimal secara relatif cepat dan dengan kinerja yang meyakinkan dibandingkan dengan heuristik di banyak domain. Dalam bab ini kami mengulas beberapa solusi EA untuk DCPS ramah lingkungan. Kami memperkenalkan tiga contoh perwakilan DCPS termasuk Pusat Data (DC), Jaringan Sensor Nirkabel (WSN), dan Jaringan Sensor Tubuh (BSN) dan membahas beberapa masalah komputasi ramah lingkungan dan solusi berbasis EA mereka.

1.1 PENDAHULUAN

Komputasi hijau umumnya mengacu pada penggunaan sumber daya yang efisien dalam komputasi bersamaan dengan meminimalkan dampak lingkungan, dan memaksimalkan kelayakan ekonomi. Mirip dengan masalah konsumsi sumber daya lainnya, tujuan komputasi hijau adalah untuk (i) menggunakan lebih sedikit bahan berbahaya, (ii) memaksimalkan efisiensi semua penggunaan sumber daya dalam sistem komputasi selama masa pakainya, dan (iii) menggunakan kembali sebanyak mungkin bahan daur ulang. sumber sebanyak mungkin dan untuk membuang apa yang tidak dapat didaur ulang secara bertanggung jawab. Bab ini berfokus terutama pada item kedua di atas, di mana teknik evolusioner digunakan untuk mencapai efisiensi energi dan komputasi yang aman dalam Sistem Fisik Cyber Terdistribusi (DCPS).

Riset dan industri terus mendorong paradigma komputasi hijau seperti membuat penggunaan komputer seefisien mungkin. Solusi semacam itu dapat diterapkan di berbagai tingkat sistem komputasi dari tingkat perangkat keras yang rendah seperti elektronik berdaya rendah hingga tingkat perangkat lunak yang tinggi seperti algoritma penjadwalan. Bab ini akan membahas teknik komputasi hijau yang hanya mengandalkan perangkat lunak manajemen beban kerja dan peralatan. Penempatan beban kerja yang sadar energi di pusat data, perutean yang sadar energi di WSN, dan manajemen daya serta siklus tugas adalah contoh solusi jenis ini. Tujuannya adalah untuk menggabungkan parameter komputasi ramah lingkungan yang

efektif dalam pengambilan keputusan untuk penetapan beban kerja dan manajemen daya node komputasi di DCPS.

Sistem komputasi terdistribusi adalah jaringan node komputasi yang berinteraksi satu sama lain untuk mencapai tujuan bersama. DCPS adalah sistem terdistribusi di mana sistem komputasi berinteraksi dengan lingkungan fisik berdasarkan informasi dari ruang fisik dan dunia maya. Kami memperkenalkan tiga contoh perwakilan DCPS termasuk Pusat Data (DC), Jaringan Sensor Nirkabel (WSN), dan Jaringan Sensor Tubuh (BSN). Komputasi hijau penting dalam sistem tersebut untuk meningkatkan skalabilitas dan keberlanjutan.

Umumnya penugasan sumber daya dalam sistem komputasi adalah masalah optimisasi kombinatorial hard diskrit NP, di mana menemukan solusi optimal (sumber daya apa yang harus ditugaskan untuk pekerjaan apa pada waktu apa) di antara serangkaian opsi layak diskrit terbatas akan memerlukan pencarian brute force. Namun pencarian brute force biasanya tidak layak karena skala sistem ini seperti di pusat data, jumlah kecil sumber daya di WSN, atau dampak kinerja dari upaya semacam itu. Perancang DCPS ramah lingkungan harus mencoba menemukan keseimbangan optimal dari tradeoff energi-latensi, di mana kinerja sistem dikorbankan untuk mendapatkan efisiensi energi. Hanya dalam kasus di mana manfaat komputasi hijau seperti peningkatan masa pakai sistem, peningkatan keandalan sistem, biaya kepemilikan yang lebih rendah, dan peningkatan keamanan lebih besar daripada biaya kinerja sistem, solusi seperti itu kemungkinan akan digunakan. Dalam domain yang kami survei, kami menunjukkan beberapa contoh di mana manfaat desain DCPS ramah lingkungan melebihi biayanya dan di mana solusi berbasis EA adalah solusi terbaik yang tersedia untuk masalah unik yang mereka hadapi saat ini.

Menemukan keseimbangan yang dijelaskan di atas menjadi lebih menantang di DCPS karena perilaku nonlinier dari banyak sistem fisik. Umumnya nonlinier ini meningkatkan waktu solusi ke ambang batas yang tidak dapat diterima jika tidak ditangani dengan elegan. Hal ini menambahkan faktor tambahan kerumitan yang harus dihadapi peneliti sebelum solusi desain hijau DCPS dapat dijalankan.

Evolutionary Algorithms (EA) adalah metode perencanaan dan pengoptimalan yang efisien dan terinspirasi dari alam berdasarkan prinsip evolusi alami dan genetika. Teknik-teknik ini terkenal karena kemampuannya untuk menemukan solusi yang memuaskan dalam waktu yang wajar. Hal ini membuat mereka sangat cocok untuk memecahkan masalah kompleks komputasi hijau di DCPS. Solusi evolusioner menemukan solusi yang hampir optimal dengan teknik pencarian acak berbasis evolusioner yang menggunakan fungsi objektif sebagai metrik. Literatur terbaru memanfaatkan kemampuan ini untuk memberikan solusi komputasi hijau yang memuaskan dengan mengatasi tantangan tersebut. Algoritma genetika, evolusioner diferensial, dan optimisasi multi-tujuan diusulkan untuk mengatasi masalah pertukaran energi-latensi di pusat data dan WSN.

1.2 KOMPUTASI HIJAU DI DOMAIN DCPS

Untuk domain menjadi DCPS, ia harus mengintegrasikan proses fisik di lingkungan dengan perangkat lunak. Pemahaman tentang interaksi antara peralatan komputasi dan lingkungan fisik dapat dimanfaatkan untuk meningkatkan efisiensi energi sistem dengan cara

yang tidak mungkin dilakukan hanya dengan pendekatan cyber. Interaksi tersebut baik disengaja demi fungsi/manajemen, atau tidak disengaja yang berasal dari interaksi alami antara sistem komputasi dan lingkungan fisik. Panas yang hilang oleh node komputasi adalah contoh interaksi yang tidak disengaja sementara pemanenan energi adalah interaksi yang disengaja dari komputasi hijau. Namun, interaksi yang tidak disengaja tidak berada di luar kendali. Solusi fisik dunia maya dapat menyadari dan mengelola interaksi yang tidak disengaja di mana pun ada manfaatnya.

Secara umum komputasi hijau mengacu pada penggunaan sumber daya yang efisien dalam komputasi bersamaan dengan meminimalkan dampak lingkungan, dan memaksimalkan kelayakan ekonomi. Dalam bab ini kami mensurvei sejumlah solusi komputasi hijau yang manfaatnya meliputi:

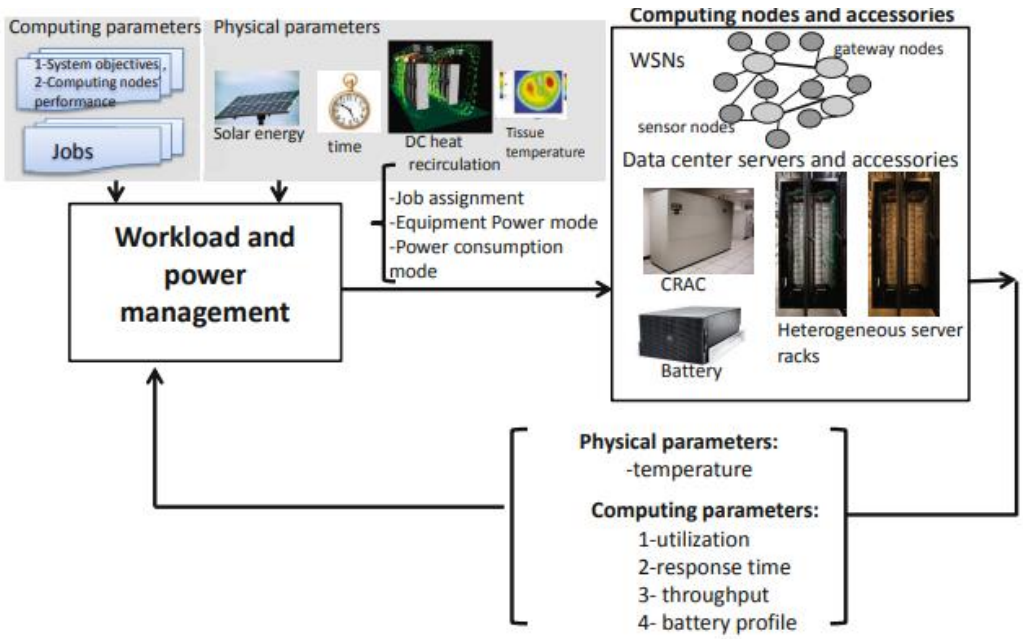
- Memaksimalkan Efisiensi Energi dengan Manajemen Beban Kerja: Dalam lingkungan yang heterogen ada potensi beberapa peralatan komputasi menjadi lebih efisien dalam komputasi daripada peralatan lainnya. Alternatifnya, bahkan di lingkungan yang homogen jika peralatan komputasi memiliki infrastruktur pendukung maka ada potensi beberapa peralatan komputasi membutuhkan dukungan yang lebih sedikit daripada yang lain. Dalam kedua kasus ini, perangkat lunak manajemen beban kerja dapat digunakan untuk meningkatkan efisiensi energi selama tujuan sistem lainnya terpenuhi.
- Memaksimalkan Pemulungan Energi dan Meminimalkan Biaya Energi untuk Keberlanjutan Biaya: Desainer hijau ingin mengais sumber energi hijau jika memungkinkan. Solusi fisik dunia maya dapat mengadaptasi beban komputasi node komputasi ke profil energi yang tersedia, atau menetapkan pekerjaan di antara node komputasi secara proporsional dengan energi hijau yang tersedia.
- Meminimalkan Bahaya pada Sistem Komputasi dan Lingkungan Fisiknya: Keamanan termal sangat penting untuk sistem komputasi apa pun. Sistem komputasi di pusat data harus berjalan di bawah suhu redline atau perangkat keras dapat rusak dan pelanggaran SLA dapat terjadi. Solusi fisik dunia maya perlu mengetahui persyaratan keamanan sistem. Contoh di mana hal ini berperan adalah di mana jadwal dapat menghindari sistem yang terlalu panas di pusat data.

Mencapai tujuan komputasi ramah lingkungan di atas dalam DCPS, yang merupakan konsumen energi besar karena skalanya yang besar atau tunduk pada ukuran baterai yang terbatas atau keduanya, adalah hal yang diinginkan. Namun, karena dinamika lingkungan fisik yang kompleks serta beberapa tujuan kinerja komputasi, solusi yang tepat dengan biaya komputasi yang rendah tidak mungkin ditemukan. Di sisa bagian ini kami memperkenalkan berbagai domain DCPS dan membahas tantangan dan manfaat penerapan solusi komputasi ramah lingkungan untuk mereka.

Pusat Data

Pusat data adalah contoh menonjol dari DCPS berdaya tinggi di mana upaya komputasi ramah lingkungan berfokus pada berbagai bentuk manajemen beban kerja. Pada dasarnya manajemen beban kerja mencoba untuk melakukan perhitungan di pusat data kapan dan di mana biayanya paling rendah dan untuk menghemat energi sebanyak mungkin pada

infrastruktur pendukung. Ada beberapa faktor yang dapat dimanfaatkan: (i) efisiensi sistem pendingin dapat ditingkatkan dengan mengatur distribusi panas antar server, (ii) efisiensi komputasi dapat ditingkatkan dengan menggunakan peralatan yang paling hemat energi di pusat data dan dengan menskalakan mode konsumsi daya server ke beban kerja input (misalnya, DVFS), dan (iii) biaya listrik dapat diturunkan dengan memanfaatkan variasi spatio-temporal biaya listrik dan ketersediaan sumber energi terbarukan. Sebuah solusi komputasi hijau untuk pusat data dapat terdiri dari kombinasi faktor-faktor tersebut seperti yang ditunjukkan oleh Gambar 1.1.



Gambar 1.1. Diagram untuk solusi fisik siber komputasi hijau di pusat data dan WSN melalui beban kerja dan manajemen daya

WSN

WSN adalah DCPS yang digunakan untuk memantau lingkungan fisik (misalnya, suhu dan kelembaban), mengirim data yang dikumpulkan ke stasiun pangkalan tempat penyimpanannya, dan menggerakkan operasi yang dikendalikan di lingkungan fisik. Tujuan desain utama dari WSN adalah keberlanjutan. Perangkat lunak manajemen biasanya dirancang untuk menjadi responsif dan eksploitatif terhadap interaksi fisik dunia maya serta peluang terkait dunia maya. Manajer perangkat lunak WSN dapat mengeksploitasi interaksi fisik dunia maya dalam beberapa cara: (i) penjadwal beban kerja dapat mengeksploitasi profil energi heterogen sensor, jenis komunikasi yang dapat diakses, dan jenis pekerjaan mereka untuk meningkatkan masa pakai sistem. (ii) pemanenan energi hijau; node sensor memiliki baterai terbatas dan masa pakai sistem dapat ditingkatkan dengan menggunakan energi hijau kapan dan di mana tersedia.

BSN

BSN pada dasarnya adalah WNS yang berspesialisasi dalam pemantauan data kesehatan. Berbeda dengan fokus pada keberlanjutan, komputasi hijau di BSN berfokus pada

meminimalkan bahaya yang disebabkan oleh DCPS pada lingkungan fisiknya (misalnya manusia). Perangkat lunak manajemen di BSN dirancang untuk meminimalkan dampak suhu dari node komputasi untuk menghindari pembakaran jaringan di sekitarnya. BSN memiliki tantangan unik karena fokus ini dan kapasitas komputasinya yang rendah.

Pernyataan Masalah untuk Green Computing di DCPS

DCPS dapat dimodelkan dengan himpunan S = s1, s2,..., si,..., sn dari node komputasi, di mana S = n, menunjukkan jumlah node. Node komputasi ditargetkan untuk melakukan beban kerja yang terdiri dari kumpulan tugas J di mana J = j1, j2,... jk,... jm . Contoh pekerjaan di DCPS adalah pekerjaan batch di pusat data komputasi kinerja tinggi (HPC), lalu lintas transaksional di pusat data web, dan perutean paket di WSN.

Komputasi Penjadwalan Beban Kerja Sadar Energi di DCPS

Skema perangkat lunak ada yang membahas komputasi hijau di DCPS dengan hanya memperhitungkan perilaku dunia maya sambil mengabaikan perilaku fisik. Skema semacam itu biasanya memperdagangkan energi untuk tujuan kinerja sistem lainnya seperti waktu respons pekerjaan. Tujuan kinerja dari sistem terdistribusi adalah kinerja yang dirasakan pekerjaan (yaitu latensi pekerjaan) atau kinerja kumulatif semua pekerjaan (misalnya, latensi rata-rata semua pekerjaan dan throughput sistem) atau keduanya. Tujuan ini biasanya tidak independen. Pengorbanan mungkin diperlukan untuk mencapai tujuan yang diinginkan masing-masing. Berbagai tujuan dalam sistem seperti itu dapat diwakili oleh vektor o yang berisi semua pekerjaan yang dirasakan tujuan kinerja o(J), tujuan kinerja yang dirasakan sistem o(S), dan tujuan energi sistem o(E). Komputasi hijau dalam lingkungan seperti itu dapat dinyatakan sebagai masalah optimisasi multi-tujuan:

$$Tujuan = optimalkan\ [o(J) dan\ o(S)\ dan\ o(E)]. \qquad (1.1)$$

Contoh pengoptimalan tersebut adalah pengoptimalan gabungan dari biaya konsumsi energi, yaitu, o(E), yang dapat menjadi nilai moneter aktual untuk biaya energi, biaya pelanggaran kinerja, yaitu, o(J) yang dapat menjadi SLA pendapatan hilang (model biaya utilitas SLA), dan biaya migrasi sistem, yaitu, o(S), di mana biaya migrasi tidak dapat disembuhkan untuk setiap perubahan status sistem karena keputusan baru.

Penjadwalan Beban Kerja Sadar Termal di DCPS

Node komputasi menghilangkan panas saat menjalankan tugas. Untuk memastikan keamanan termal, suhu node komputasi tidak boleh melebihi suhu operasi aman yang ditentukan oleh pabrikan atau bergantung pada lingkungan tempat mereka ditempatkan. Biarkan T menjadi vektor suhu node. Masalah penjadwalan dan komunikasi yang menyadari keselamatan termal berurusan dengan pendistribusian tugas, J, di antara n node sedemikian rupa sehingga kondisi keamanan termal dari semua node terpenuhi. Masalah ini diperumit oleh tradeoff energi-latensi yang biasa terjadi dalam desain sistem dan khususnya rumit dalam DCPS karena melibatkan model fisik kompleks, non-Markovian. Misalnya, di pusat data, simpul diam yang telah diam dalam jangka waktu lama akan memanas lebih lambat daripada simpul diam yang

baru saja berhenti menjalankan tugas. Penempatan beban kerja sadar keselamatan termal membutuhkan meminimalkan kasus terburuk (titik panas). Secara formal masalah ini dinyatakan sebagai berikut:

$$\begin{aligned} &Minimize\ maximize_i\ T_i \\ &Subject\ to \qquad T_i = f, \forall i \\ &\qquad\qquad J\ ditugaskan\ dan\ batasan\ kinerja\ J\ terpenuhi, \end{aligned} \tag{1.2}$$

Dimana f adalah fungsi yang menghubungkan suhu setiap node dengan tugas pekerjaan dari node itu sendiri dan semua node lainnya. Pertanyaannya adalah bagaimana menugaskan pekerjaan ke setiap node. Secara algoritmik, ini mirip dengan masalah Alokasi Sumber Daya Min-Max (MMRA) yang dikenal sebagai NP lengkap. Kompleksitas masalah berasal dari skala node serta ketergantungan antar node untuk kinerja dan pembuangan panas.

Pemanenan Energi dan Manajemen Biaya

Biaya energi dapat menunjukkan nilai moneter aktual per unit energi atau jenis energi yaitu hijau atau coklat (energi hijau menunjukkan energi terbarukan seperti matahari sedangkan energi coklat menunjukkan jenis energi lain seperti nuklir). Tujuan dari manajemen biaya energi adalah untuk menyediakan komputasi yang berkelanjutan dengan mengkonsumsi energi berbiaya rendah. Untuk mencapai keberlanjutan, teknik manajemen daya sadar biaya diperlukan untuk menyeimbangkan beban dan kinerja node DCPS dengan mencocokkan konsumsi energi spatio-temporal DCPS dengan profil energi berbiaya rendah yang tersedia. Solusi umum untuk keberlanjutan dapat dinyatakan sebagai berikut: (i) penyangga energi dalam baterai setiap kali energi berbiaya rendah tersedia dan mengalirkan energi keluar dari baterai saat tidak ada, (ii) simpul perhitungan siklus kerja untuk menyesuaikan konsumsi energi dengan profil energi yang tersedia, dan (iii) memigrasikan pekerjaan ke node di mana energi berbiaya rendah tersedia. Kombinasi operasi ini dapat mengatur penggunaan energi DCPS untuk meminimalkan biaya energi. Namun, dalam praktiknya, solusi harus mengatasi tantangan berikut: (i) ketidakpastian yang terkait dengan ketersediaan energi berbiaya rendah, (ii) overhead dan kendala migrasi pekerjaan, (iii) pelanggaran persyaratan kinerja pekerjaan dari siklus tugas, dan (iv) lonjakan beban kerja yang sifatnya tidak dapat diprediksi yang mencegah siklus tugas yang optimal.

Untuk mendefinisikan masalah secara formal, misalkan quadruple (si, jk, v, t) menjadi vektor penugasan pekerjaan yang menyatakan pekerjaan jk pada waktu t ditugaskan ke simpul si yang berjalan di bawah mode konsumsi daya v atau kebijakan siklus tugas. Masalah optimisasi biaya energi dapat dimodelkan sebagai sistem waktu diskrit, dimana pada setiap waktu t, masalah tersebut menentukan penugasan pekerjaan dan sumber konsumsi daya dari node komputasi. Energi berbiaya rendah dan beban kerja dalam banyak kasus menunjukkan perilaku musiman (misalnya, perilaku beban kerja Internet siklik), sehingga masalah optimalisasi biaya dapat

dikelola dalam rentang waktu TW, di mana ketersediaan energi ramah lingkungan dan beban kerja dapat diprediksi sebagai berikut:

$$minimize \sum_{t=1}^{Tw} \sum_{i=1}^{n} \sum k = 1^{m} EnergyCost(s_i, j_k, v, t), B_i, p_i^{AC}$$

Subject Energi berbiaya rendah (1.3)
Kendala penugasan pekerjaan
Kendala kinerja pekerjaan
Batasan migrasi pekerjaan
Node tersedia mode manajemen daya/tugas,

Dimana *EnergyCost* adalah fungsi yang memetakan vektor penugasan pekerjaan ke biaya energi menurut profil energi berbiaya rendah yang tersedia di setiap node (dilambangkan dengan B_i) dan biaya sumber energi lainnya dilambangkan dengan P_i^{AC}. Kompleksitas masalahnya adalah dua kali lipat: (i) sifat nonlinier dari fungsi manajemen daya atau fungsi siklus tugas (misalnya, skala voltase diskrit CPU), dan (ii) biaya konsumsi daya nonlinier. Biaya konsumsi daya dari node komputasi terdiri dari daya diam dan daya dinamis. Daya dinamis adalah fungsi dari penugasan pekerjaan sementara daya diam tidak bergantung pada penugasan pekerjaan. Kompleksitas masalah diperparah ketika sifat stokastik energi murah diperhitungkan.

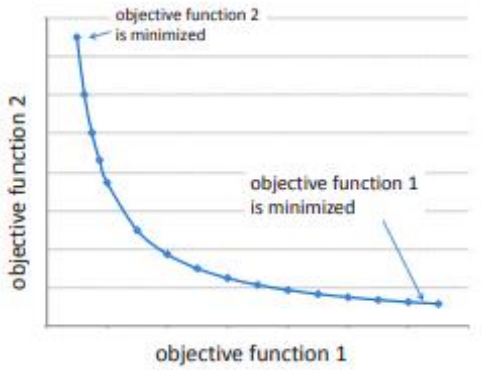
Contoh masalah seperti itu di pusat data adalah beban kerja dan manajemen server di seluruh pusat data, di mana tujuannya adalah untuk mendistribusikan beban kerja di seluruh pusat data sesuai dengan biaya dan jenis listrik saat ini (hijau/coklat). Dalam WSN mengelola pemanenan energi untuk keberlanjutan adalah contoh dari masalah ini.

1.3 TINJAUAN TENTANG EA

Ada berbagai jenis algoritma evolusioner di mana gagasan umum yang mendasarinya adalah sebagai berikut: diberikan populasi individu, seleksi acak terpandu menyebabkan seleksi alam (yaitu, *survival of the fittest*) dan ini menyebabkan peningkatan kebugaran populasi untuk mencapai tujuan yang diinginkan. EA ditargetkan untuk mengoptimalkan fungsi metrik. Untuk itu, algoritme membuat sekumpulan kandidat solusi secara acak dan menerapkan fungsi metrik sebagai ukuran kebugaran. Berdasarkan kesesuaian ini, beberapa kandidat yang lebih baik dipilih untuk menyemai generasi berikutnya yang dihasilkan oleh rekombinasi/mutasi dari generasi saat ini. Proses ini diulang sampai solusi dengan kualitas yang diinginkan ditemukan atau batas komputasi tercapai. Proses seperti itu sangat efektif dalam memecahkan masalah optimisasi yang kompleks karena alasan berikut: Pertama, penggunaan populasi solusi membantu EA menghindari "terperangkap" pada optimal lokal, ketika optimal yang lebih baik dapat ditemukan di luar sekitar solusi saat ini. Kedua, kompleksitas fungsi tujuan tidak menambah kompleksitas proses, karena model tidak perlu mengetahui sifat masalah dan solusi yang layak.

Kelemahan utama EA adalah tidak memiliki konsep “solusi optimal”. Sebuah solusi "lebih baik" hanya dibandingkan dengan solusi "yang diketahui saat ini". Untuk alasan ini, EA digunakan pada masalah yang sulit atau tidak mungkin untuk menguji optimalitasnya. Ada berbagai kelas solusi EA. Kami meninjau tiga kelas EA berikut yang digunakan dalam masalah komputasi hijau.

- *Genetic Algorithms* (GA): GA memodelkan evolusi genetik dan terdiri dari proses utama berikut: (i) inisialisasi di mana populasi awal dihasilkan secara acak atau diunggulkan menuju area di mana solusi optimal mungkin ditemukan, (ii) seleksi dimana sebagian populasi dipilih untuk menghasilkan generasi berikutnya. Solusi individu biasanya dievaluasi melalui fungsi kebugaran, (iii) reproduksi di mana generasi berikutnya dari populasi diproduksi melalui operator genetik: persilangan (juga disebut rekombinasi), dan/atau mutasi. Untuk setiap solusi baru yang dihasilkan, yaitu anak, sepasang atau beberapa solusi “induk” dipilih. Metode crossover dan mutasi di atas, kemudian digunakan untuk membuat solusi baru yang biasanya memiliki banyak kesamaan dengan karakteristik “orang tua” mereka, akhirnya (v) terminasi: proses generasi (misalnya, prosedur seleksi dan reproduksi) diulang sampai kondisi penghentian, misalnya, batas waktu atau kesesuaian yang memadai telah tercapai.
- Evolusi Diferensial (DE): DE digunakan untuk fungsi bernilai riil multidimensi. Ini identik dengan GA kecuali untuk mekanisme reproduksinya. Ide inti DE adalah mengadaptasi langkah pencarian secara inheren sepanjang proses evolusi dengan cara yang memperdagangkan eksploitasi dan eksplorasi.
- Algoritma genetika tujuan ganda (MOGA): Sementara masalah optimisasi tujuan tunggal memiliki solusi optimal yang unik, masalah tujuan ganda biasanya memiliki sekumpulan solusi yang dikenal sebagai himpunan optimal Pareto. Bayangan himpunan ini dalam ruang objektif dinotasikan sebagai front Pareto seperti yang ditunjukkan pada Gambar 1.2. MOGA mengadaptasi operator evaluasi dan seleksi GA untuk memungkinkan konvergensi proses evolusi ke depan Pareto terbaik dan untuk mempertahankan beberapa keragaman solusi potensial. Strategi serupa diperlukan dalam evolusi diferensial multi-tujuan.



Gambar 1.2. Contoh graf depan Pareto

- *Swarm Intelligence* (SI): SI adalah teknik optimasi stokastik berbasis populasi, terinspirasi oleh perilaku sosial spesies serangga tertentu. Ini memanfaatkan perilaku

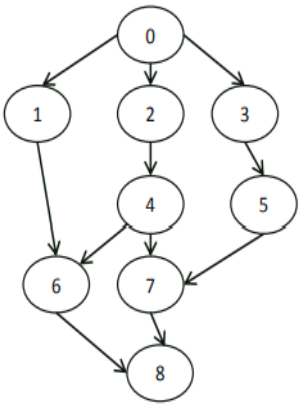
kompleks dan seringkali cerdas melalui interaksi kompleks dari ribuan anggota kawanan otonom. Interaksi semacam itu didasarkan pada naluri primitif yang dilakukan dalam komunikasi melalui lingkungan. Contohnya adalah feromon yang digunakan semut untuk berkomunikasi. Feromon adalah zat kimia yang dapat dirasakan oleh semut saat berjalan di sepanjang jalur. Semut tertarik dengan feromon dan cenderung mengikuti jalur dengan konsentrasi feromon yang tinggi. Konsentrasi feromon meningkat ketika semut yang tertarik berbaring lebih banyak pada jalur yang sama. Ant Colony Optimization (ACO) adalah kelas SI yang bekerja dengan mensimulasikan perilaku semut yang menyimpan feromon dan mengikuti feromon. Mengingat hal ini, fungsi algoritma ACO dapat diringkas sebagai berikut. Seperangkat komputasi bersamaan dan agen asinkron (koloni semut) bergerak melalui status masalah yang sesuai dengan solusi perantara dari masalah yang akan dipecahkan. Langkah mereka didasarkan pada kebijakan keputusan lokal stokastik yang berasal dari dua parameter, jejak dan daya tarik (visibility). Nilai jejak diperbarui secara iteratif ketika seekor semut menyelesaikan solusi yang mengarahkan pergerakan semut di masa depan.

1.4 APLIKASI EA UNTUK *GREEN COMPUTING* DI DCPS

Semua masalah yang disebutkan di atas adalah masalah optimisasi kombinatorial yang sulit diselesaikan, yang solusi optimalnya tidak dapat dicapai dengan cara komputasi yang efisien. Untuk alasan ini, beberapa literatur mengusulkan untuk menggunakan solusi heuristik dan metaheuristik termasuk solusi berbasis EA untuk menyelesaikannya. Pada bagian ini kami meninjau studi kasus yang menunjukkan bagaimana solusi berbasis EA digunakan untuk memecahkan masalah terkait komputasi hijau di pusat data dan WSN. Batas kinerja teoretis untuk solusi EA jarang ditemukan dalam praktiknya. Sebelum kita dapat membahas kinerja skema yang diusulkan, pertama-tama kita perlu memberikan gambaran singkat tentang solusi heuristik. Kami kemudian membahas solusi berbasis EA secara lebih rinci dalam hal (i) bagaimana masalah komputasi hijau dimodelkan menggunakan skema EA yang ada, dan (ii) bagaimana kinerja solusi EA dibandingkan dengan solusi heuristik.

Survei Solusi Berbasis Evolusioner untuk Penjadwalan Beban Kerja Sadar Energi di Pusat Data Komputasi Kinerja Tinggi (HPC)

Pusat data HPC berorientasi pada aplikasi intensif komputasi paralel, mis. simulasi atau pemrosesan kumpulan data besar, yang membutuhkan banyak prosesor dan dapat berjalan dalam jangka waktu lama. Penjadwalan pekerjaan pusat data HPC didefinisikan sebagai proses mengalokasikan set J dari m tugas ke set S dari n mesin di mana setiap prosesor diizinkan untuk beroperasi pada level voltase yang berbeda. Pekerjaan yang ditetapkan J dapat menunjukkan pekerjaan dependen atau independen. Yang terakhir mungkin memberlakukan prioritas pada eksekusi pekerjaan sehubungan dengan ketergantungan (lihat Gambar 1.3). Tujuannya adalah meminimalkan kekuatan atau mendapatkan tradeoff yang diinginkan di antara semua tujuan sistem dan pekerjaan (misalnya, makspan, QoS, dan kekuatan).



Gambar 1.3. DAG mewakili serangkaian tugas yang bergantung

Untuk mengelola energi, komunitas riset memasukkan DVFS ke dalam penjadwalan tugas. Aplikasi model ilmiah dengan Directed Acyclic Graphs (DAG) (lihat Gambar 1.3) dan mengusulkan untuk mengurangi konsumsi energi dengan menurunkan tegangan pasokan prosesor selama pelaksanaan tugas-tugas non-kritis. Algoritma menggunakan metrik "Keunggulan Relatif (RA)" yang mengukur efisiensi energi relatif antara setiap dua tugas tugas dan prosesor serta level voltase yang sesuai. Pertama-tama, algoritme mencari penugasan tugas yang layak dan perintah eksekusi dan kemudian menggunakan metrik RA untuk memilih penugasan tugas yang paling hemat energi dari solusi yang layak.

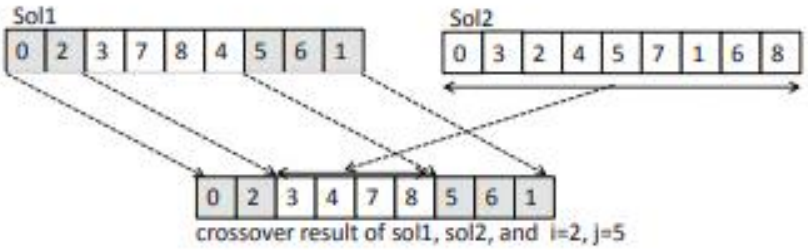
Pertimbangkan tiga parameter berikut untuk penjadwalan tugas: tugas *j*, prosesor *p*, dan voltase *v*. RA dari *(j, p, v)* dan *(j, p', v')* dihitung sebagai berikut:

$$RA(j,p,v,p',v') = -\left[\left[\frac{E(j,p,v) - E(j,p',v')}{E(j,p,v)}\right] + \left[\frac{EFT(j,p,v) - EFT(j,p'v')}{EFT(j,p,v) - \min\left(EST(j,p,v), EST(j,p'v')\right)}\right]\right] \quad (1.4)$$

$E(j,p,v)$ menunjukkan konsumsi energi total untuk mengeksekusi pekerjaan *j* pada prosesor *p* dan voltase *v*. Juga $EFT(j,p,v)$, dan $EST(j,p,v)$ menunjukkan penyelesaian paling awal waktu dan waktu mulai paling awal dari pekerjaan j masing-masing pada prosesor dan voltase yang diberikan. Ada beberapa pekerjaan yang menggunakan solusi berbasis EA seperti GA, ACO, dan GA multi-objektif untuk memecahkan masalah penjadwalan tugas sadar energi. Deskripsi singkat tentang solusi berbasis EA tersebut dan evaluasi kinerjanya diberikan di bawah ini.

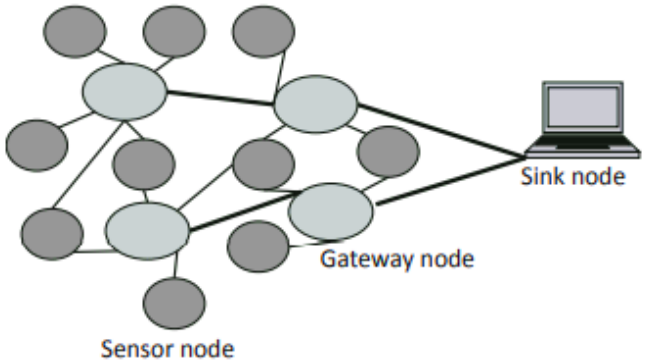
Penelitian dari Mezmaz yang menyempurnakan skema heuristik sebelumnya ECS dengan membuat hibrida GA dan ECS yang dijelaskan di bawah ini. GA digunakan untuk menyediakan penjadwalan tugas yang layak, dan ECS digunakan untuk memberikan tugas tugas sadar energi. Sebuah kromosom, yaitu, solusi, terdiri dari gen N di mana setiap gen adalah solusi penjadwalan untuk tugas, prosesor, dan penskalaan tegangan pada prosesor, misalnya, (j,p,v). Output dari GA adalah kumpulan tugas optimal Pareto di mana batasan prioritas dihormati. Dengan kata lain, GA membangun bagian tugas dari sebuah solusi. Operator crossover dalam GA yang diusulkan menggunakan dua solusi untuk menghasilkan

solusi baru. Asumsikan dua solusi pemesanan tugas, sol_1 dan sol_2. Operator penyilangan menghasilkan sol'_1 dan $sol'*$ dengan mencampurkan solusi asli sehingga prioritas terpenuhi. Misalnya untuk menghasilkan sol'_1, memilih dua bilangan bulat acak x dan y, sehingga $0 \leq x \leq y \leq |J| = m$, dan semua tugas sol_1 sebelum x dan setelah y disalin ke sol'_1 secara berurutan. Kemudian semua tugas sol_2 yang belum ada di sol'_2 disalin ke posisi sol'_1 yang terletak di antara x dan y secara berurutan (lihat Gambar 1.4). Ketika solusi parsial sudah siap, ECS dipanggil untuk menghitung metrik RA dari tugas yang diberikan pada semua prosesor dan penskalaan voltase. ECS membangun bagian prosesor dan voltase yang tersisa dari solusi parsial (yaitu, p dan v) yang disediakan oleh operator mutasi dan persilangan GA. Operator kebugaran GA dipanggil setelah prosesor tugas dan bagian voltase dari setiap gen solusi diketahui. Peran operator ini adalah menghitung konsumsi energi dan masa pakai setiap solusi. Selain itu, penulis mengusulkan heuristik, multi-bintang, yang menggunakan paralelisasi untuk mengurangi waktu eksekusi skema GA yang diusulkan.



Gambar 1.4. Operasi crossover dari algoritma penjadwalan berbasis GA seperti yang diusulkan dalam

Skema yang diusulkan telah diimplementasikan menggunakan ParadisEO. Platform perangkat lunak ini menyediakan alat untuk desain meta-heuristik paralel untuk pengoptimalan multi-tujuan. Eksperimen telah dilakukan pada kisi tiga kluster yang berisi total 714 inti. Pendekatan tersebut telah dievaluasi dengan grafik tugas Fast Fourier Transformation yang merupakan aplikasi dunia nyata. Eksperimen menunjukkan meta-heuristik berbasis GA yang diusulkan mengurangi konsumsi energi sebesar 47,5% dan waktu penyelesaian sebesar 12% dibandingkan dengan solusi penjadwalan berbasis meta heuristik lainnya. Selain itu, pendekatan multi-bintang rata-rata 13 kali lebih cepat daripada pendekatan paralelisasi GA yang diusulkan sebelumnya.



Gambar 1.5. WSN berbasis Gateway (cluster).

Penelitian yang dilakukan Shen juga menggunakan GA untuk memecahkan masalah penjadwalan tugas sadar energi dengan menggunakan DVFS pada set pekerjaan independen. penulis mengusulkan Shadowed GA (SGA) untuk mengatasi kecepatan pencarian GA yang rendah. SGA menggunakan sistem dua pengukuran: nilai kebugaran (seperti konsumsi energi total) digunakan untuk mengevaluasi solusi keseluruhan (kromosom) dan harga bayangan digunakan untuk mengevaluasi komponen solusi (gen). Dengan cara ini, harga bayangan secara relatif mengukur nilai kecocokan solusi dengan setiap perubahan komponen. Intuisi di balik penggunaan harga bayangan membuat operasi GA lebih cerdas. GA dapat menggunakan harga bayangan untuk secara langsung membandingkan komponen, hubungannya, dan kontribusinya terhadap solusi yang lebih baik. Untuk penjadwalan tugas sadar energi, harga bayangan didefinisikan sebagai konsumsi energi rata-rata per instruksi untuk setiap prosesor. SGA menambahkan satu operasi lagi ke operasi yang disediakan oleh mutasi GA klasik dan operasi persilangan. Misalnya untuk memilih subpopulasi, salah satu operasi berikut dilakukan secara acak:

- Operasi mutasi klasik (Bergerak). Pilih dua prosesor secara acak dan pindahkan satu tugas yang dipilih secara acak dari satu prosesor ke prosesor lainnya.
- Operasi crossover klasik. Tukarkan dua tugas yang dipilih secara acak antara dua prosesor yang dipilih secara acak.
- Operasi mutasi terpandu harga bayangan. Mutasi dan tukar tugas sesuai dengan harga yang dibayangi.

Menggunakan operasi di atas evolusi akan mengurangi harga bayangan prosesor dengan memindahkan tugas di antara mereka. Penulis menunjukkan bahwa SGA menghemat lebih banyak energi daripada GA dan dapat menemukan solusi dalam waktu yang lebih singkat daripada GA dalam studi simulasi mereka.

Algoritma penjadwalan dirancang untuk aplikasi alur kerja di mana beberapa parameter kinerja yang ditentukan pengguna diinginkan seperti keandalan, waktu, dan biaya. Parameter biaya dapat diartikan sebagai biaya energi; oleh karena itu pekerjaan ini dapat dimanfaatkan untuk manajemen biaya energi. Penulis mengusulkan beberapa heuristik untuk mengoptimalkan tujuan kinerja dan skema adaptif dirancang untuk memungkinkan semut buatan memilih heuristik berdasarkan nilai feromon. Kinerja algoritma dibandingkan dengan tenggat waktu-MDP yang hanya dapat menangani tugas optimalisasi biaya dengan batasan tenggat waktu. Hasilnya menunjukkan algoritma ACS berhasil menurunkan biaya sebesar 10-20% dibandingkan dengan pendekatan Deadline-MDP.

Rangkuman Hasil

Studi ini menunjukkan bahwa solusi berbasis EA dapat menghasilkan kinerja yang lebih tinggi daripada solusi heuristik. Literatur menunjukkan bahwa kinerja EA dapat ditingkatkan secara signifikan dengan paralelisasi dan dengan memasukkan heuristik khusus aplikasi dalam pemodelan. Hasilnya menunjukkan bahwa solusi berbasis EA dapat diterapkan pada domain penjadwalan. Namun untuk memutuskan penerapannya dalam praktik, studi komprehensif yang membandingkan solusi heuristik dengan solusi EA dan menganalisis tradeoff biaya-kinerja yang terlibat akan berguna. Penelitian saat ini tidak memiliki evaluasi ini.

Survei Masalah Rute Hemat Energi untuk WSN

Pertimbangkan satu set S node sensor terhubung yang didistribusikan secara acak di wilayah di mana setiap node memiliki masa pakai baterai terbatas yang digunakan terutama untuk mentransmisikan data. Setiap node mengumpulkan data yang perlu dikirimkan ke node gateway yang memiliki daya pemrosesan lebih besar dan jangkauan transmisi lebih besar. Perutean hemat energi dalam sistem seperti itu biasanya dirancang untuk memaksimalkan masa pakai sistem. Perhatikan bahwa solusi energi minimum tidak sama dengan solusi seumur hidup maksimum karena jika semua lalu lintas diarahkan melalui jalur yang paling hemat energi, node di jalur tersebut akan habis dengan cepat. Dalam WSN, lebih penting untuk merutekan lalu lintas sedemikian rupa sehingga konsumsi energi seimbang di antara node sebanding dengan energi yang tersedia daripada meminimalkan konsumsi energi total. Oleh karena itu, pernyataan masalah umum untuk masalah perutean hemat energi adalah sebagai berikut: Diberikan satu set rute yang layak, untuk setiap node, temukan rute dan frekuensi yang harus digunakan setiap rute untuk memaksimalkan masa pakai node. Secara matematis, masalah routing seperti itu diformalkan sebagai pemrograman integer dan NP-complete. Node WSN perlu membuat keputusan cepat untuk memastikan overhead yang rendah. Untuk alasan itu, heuristik dan meta-heuristik diusulkan untuk menemukan rute.

Beberapa contoh solusi heuristik adalah sebagai berikut. Heinzelman dkk. mengusulkan algoritma Low *Energy Adaptive Clustering Hierarchy* (LEACH) yang mungkin merupakan salah satu protokol yang paling banyak direferensikan di area jaringan sensor. Algoritme secara merata menyeimbangkan beban energi di antara node sensor dengan menggunakan rotasi acak stasiun basis cluster lokal. *Power-Efficient Gathering inSensor Information Systems* (PEGASIS) adalah skema heuristik lain yang meningkatkan manfaat hemat energi dari LEACH dengan memaksa setiap node berkomunikasi hanya dengan tetangga dekat dan bergiliran mentransmisikan ke stasiun pangkalan. Satu set jalur sub-optimal kadang-kadang untuk meningkatkan umur jaringan. Jalur ini dipilih melalui fungsi probabilitas yang bergantung pada konsumsi energi setiap jalur. Gupta dkk. mengusulkan sebuah algoritma untuk membuat jaringan sensor menjadi cluster sedemikian rupa sehingga node gerbang yang dibatasi energi lebih sedikit bertindak sebagai clusterhead. Beban kemudian diseimbangkan di antara gerbang-gerbang ini.

Ada juga beberapa solusi berbasis EA untuk masalah perutean hemat energi dalam literatur termasuk ACO, DE, GA. Algoritma Routing Kolaboratif yang terinspirasi semut untuk Umur Panjang Jaringan Sensor Nirkabel (CRAWL) yang terbukti memiliki kinerja tinggi ketika node memiliki pasokan energi yang tidak seragam.

Algoritma *Energy Efficient Ant Based Routing* (EEABR) yang menggunakan semut buatan untuk menemukan jalur routing antara node sensor dan sink node, yang dioptimalkan dalam hal jarak dan tingkat energi. Untuk tujuan ini, penulis mendefinisikan fungsi visibilitas sebagai kebalikan dari konsumsi energi total setiap node, dan jejak pheromone sebagai tingkat lalu lintas dalam tabel routing di setiap node. Feromon jejak dihitung untuk menunjukkan tingkat energi dan panjang jalur. Oleh karena itu, probabilitas pemilihan node ditentukan oleh tradeoff antara visibilitas ACO yang merupakan keinginan kita untuk menggunakan node dengan lebih banyak energi dan intensitas jejak aktual yang merupakan keinginan kita untuk

meminimalkan energi transmisi. Skema ini dibandingkan dengan skema perutean ACO penulis lainnya yang mengabaikan visibilitas di atas, dan hasilnya menunjukkan bahwa kinerja EEABR lebih unggul.

Pendekatan yang berbeda dalam memanfaatkan ACO untuk mencapai efisiensi energi dalam masalah perutean WSN. Mengadopsi model deret waktu, ARMA, untuk menganalisis kecenderungan dinamis dalam lalu lintas data dan untuk menyimpulkan konstruksi faktor beban, yang dapat membantu mengungkap status energi masa depan sensor dalam WSN. Faktor beban kemudian digunakan untuk memandu aturan pembaruan feromon sedemikian rupa sehingga semut buatan meramalkan keadaan energi jaringan lokal dan kemudian tindakan yang sesuai diambil secara adaptif untuk meningkatkan efisiensi energi dalam konstruksi perutean. Hasil simulasi menunjukkan bahwa skema ACO yang diusulkan memang dapat menyeimbangkan total biaya energi untuk transmisi data.

Masalah perutean untuk WSN di mana semua node mampu mencapai satu sama lain, menggunakan pendekatan *Multi-Objective Differential Evolution* (MODE) untuk mengurangi energi serta penundaan transmisi di WSN. Skema yang diusulkan didasarkan pada redaman daya nonlinier di mana daya sinyal gagal dengan faktor $r-\alpha$, di mana r menunjukkan jarak dari pengirim dan α menunjukkan faktor redaman daya. Karena pelemahan daya nonlinear ini, menyampaikan sinyal menggunakan node perantara dapat menghasilkan konsumsi daya yang lebih rendah daripada komunikasi langsung dengan potensi biaya latensi.

Oleh karena itu, satu set perutean optimal Pareto sehubungan dengan beberapa tujuan (yaitu, mengurangi energi dan latensi) dapat diidentifikasi, di mana beberapa kandidat mewakili kemungkinan pertukaran yang berbeda antara konsumsi energi dan latensi komunikasi. Penulis mengusulkan versi MODE diskrit di mana variabel keputusan diwakili oleh variabel vektor multi-dimensi misalnya, $x = [x_1, x_2,..., x_n]$, $x_i \in \chi_i$, di mana χ_i adalah sekumpulan nilai diskrit yang membatasi kemungkinan nilai yang dapat dimiliki oleh xi yang bersesuaian. Karena vektor diferensial DE tradisional tidak lagi dapat ditafsirkan untuk kasus diskrit, operator probabilistik diperlukan. Operator ini termasuk probabilitas serakah, mutasi, dan perturbasi. Untuk memodelkan masalah routing penulis mendefinisikan kromosom sebagai jalur dari node sumber ke node tujuan. Jalur ini direpresentasikan sebagai urutan ID node jaringan dengan node sumber di lokus pertama dan node tujuan di lokus terakhir. Panjang kromosom bervariasi tergantung pada node sumber, node tujuan, dan jalur. Untuk memfasilitasi pencarian evolusioner, operator ekspansi diperkenalkan untuk menghasilkan populasi awal secara sistematis dan mengimplementasikan operasi reproduksi yang menggunakan seperangkat aturan heuristik berdasarkan hubungan kekuatan di antara node terkait.

GA untuk memaksimalkan umur panjang jaringan dalam hal waktu hingga kematian node pertama. Tujuannya adalah untuk menghasilkan jadwal transmisi yang terdiri dari kumpulan putaran transmisi. Jadwal transmisi menunjukkan bagaimana data dikumpulkan dari setiap sensor dan disebarkan ke stasiun pangkalan. Ini mewakili kumpulan jalur perutean yang akan diikuti jaringan untuk memaksimalkan masa pakainya. Setiap jalur perutean adalah pohon yang disebut pohon agregasi. Jalur yang paling efisien bukanlah jawaban terbaik, karena terus menggunakan jalur ini akan menyebabkan beberapa node mati lebih awal dari

yang lain. Oleh karena itu, penulis menemukan kumpulan pohon agregasi, dimana setiap pohon digunakan untuk jumlah putaran yang tetap, sehingga konsumsi energi seimbang di antara semua node dalam jaringan. Solusi adalah sekumpulan pohon agregasi dan frekuensinya. Oleh karena itu, himpunan solusi diberikan kepada GA. Jadwal transmisi adalah frekuensi semua pohon agregasi yang mewakili sebuah kromosom dan setiap gen sesuai dengan frekuensi pohon agregasi tertentu. Frekuensi pohon agregasi direpresentasikan sebagai bit dari nilai binernya. Terakhir, fungsi kebugaran dirancang untuk meningkatkan masa pakai sistem. Hasil simulasi menunjukkan bahwa algoritma GA menemukan solusi yang mendekati optimal lebih cepat daripada model program linier dari masalah tersebut.

Perutean hemat energi untuk WSN diusulkan perutean dua tingkat. Pengumpulan beban kerja dua tingkat dalam jaringan sensor, di mana ada beberapa node relai daya yang lebih tinggi yang dapat membentuk jaringan di antara mereka sendiri untuk merutekan data menuju stasiun pangkalan. Kemudian mereka mengusulkan solusi berbasis GA untuk menjadwalkan pengumpulan data node relai. Masalahnya dimodelkan sedemikian rupa sehingga setiap kromosom dalam populasi awal sesuai dengan skema perutean yang valid yang diwakili oleh nomor simpul relai. Nilai kebugaran untuk setiap individu dihitung sebagai daya awal dari setiap node relai atas energi maksimum yang hilang pada semua node relai untuk skema perutean yang ditentukan oleh kromosom. Pemilihan individu dilakukan dengan menggunakan metode pemilihan Roulette-Wheel dan operasi crossover yang seragam. Mutasi dilakukan sedemikian rupa sehingga node yang menghilangkan energi maksimum karena transfer data dipilih sebagai node kritis. Kemudian rute yang menggunakan node kritis diubah untuk mengurangi bebannya. Node alternatif untuk setiap node kritis berada dalam jangkauan transmisi dari node sumber sehingga jarak Euclidean antara sumber dan tujuan akhir lebih besar daripada jarak antara node sumber dan node alternatif. Melalui studi eksperimental, penulis menunjukkan perpanjangan waktu hidup rata-rata 200% dibandingkan dengan jaringan yang menggunakan model energi transmisi minimum dan model perutean hop minimum.

Rangkuman Hasil

Solusi berbasis EA dapat menemukan solusi yang lebih baik dibandingkan dengan solusi heuristik yang ada. Juga, berbagai solusi berbasis EA dimanfaatkan dalam literatur, tetapi tidak ada penelitian yang membandingkan kinerja relatifnya. Juga tidak ada studi berbasis evaluasi yang membandingkan pengorbanan kinerja biaya dari solusi berbasis EA dengan solusi heuristik.

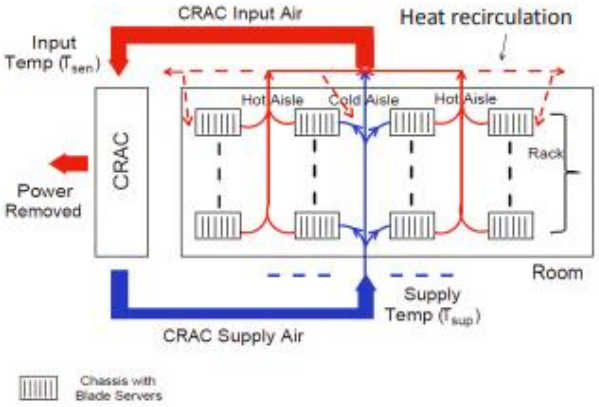
Survei Aplikasi EA untuk Penjadwalan Pekerjaan Sadar Termal di Pusat Data HPC

Penjadwalan pekerjaan sadar termal untuk pusat data HPC telah diusulkan dalam beberapa pekerjaan untuk meningkatkan efisiensi energinya. Gagasan utamanya adalah untuk meningkatkan efisiensi infrastruktur pendukung seperti CRAC dan menyediakan beban kerja pada server yang paling hemat energi. Dalam penelitian terkait, Moore dkk, dan Bash dan Forman menunjukkan bahwa penempatan beban kerja sadar termal dapat menghemat energi di pusat data. Tang dkk. dan Mukherjee dkk. memodelkan resirkulasi panas dan mengusulkan solusi berbasis heuristik dan GA untuk penjadwalan pekerjaan sadar termal spatio dan spatio-temporal. Bagian ini pertama-tama memberikan ikhtisar konsep penjadwalan sadar termal di

pusat data dan kemudian meninjau heuristik terkait dan solusi berbasis GA yang diusulkan dalam literatur.

Penjadwalan Pekerjaan Sadar Termal untuk Pusat Data: Gambaran Umum

Penjadwalan pekerjaan sadar termal di pusat data berkaitan dengan minimalisasi sirkulasi ulang panas di antara server. Gambar 1.6 menunjukkan tata letak pusat data tipikal. Cacat dalam tata letak ini menyebabkan panas bersirkulasi antar server alih-alih mengalir ke *Computer Room Air Conditioners* (CRAC). Resirkulasi panas di antara server berasal dari tata letak fisik pusat data kontemporer di mana server komputasi diatur dalam deretan rak dan susunan rak sedemikian rupa sehingga panel depan atau belakang dari setiap dua rak dalam baris saling berhadapan. Dalam pengaturan ini, disebut pengaturan lorong panas / lorong dingin, aliran udara membuat resirkulasi udara panas dari saluran keluar udara server komputasi ke udara masuknya.



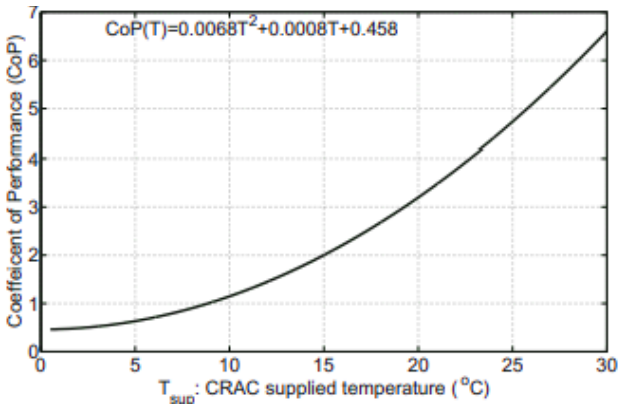
Gambar 1.6. Demonstrasi pengaturan server lorong dingin-panas dan resirkulasi panas di dalam ruang pusat data

Sirkulasi ulang panas memaksa operator pusat data untuk mengoperasikan AC ruang komputer (CRACs) mereka untuk memasok udara dingin, dilambangkan sebagai T_{sup}, pada suhu yang jauh lebih rendah daripada suhu redline server, T_{red}, yang merupakan suhu operasi aman maksimum dari server. Besarnya resirkulasi panas bergantung pada tata letak fisik/aliran udara ruangan pusat data. Server tidak memberikan kontribusi yang sama terhadap resirkulasi panas. Resirkulasi panas antar server dimodelkan sebagai matriks N oleh N, D, di mana setiap elemen D, d_{ij}, adalah rasio suhu keluaran server j terhadap suhu masuk server i. Biarkan p, menjadi vektor kekuatan dari semua server dan T_{in} menjadi vektor suhu masuk dari semua server sedemikian rupa sehingga:

$$T_{in} = T_{sup} + Dp. \qquad (1.5)$$

Karena beban kerja langsung mempengaruhi daya, tugas beban kerja secara langsung mempengaruhi distribusi suhu. Energi pendinginan CRAC dapat dimodelkan dengan koefisien kinerjanya (CoP), yang merupakan rasio panas yang dihilangkan (yaitu, energi

komputasi) terhadap energi yang dibutuhkan untuk menghilangkan panas itu (yaitu, energi pendinginan). CoP yang lebih tinggi berarti pendinginan yang lebih hemat energi, karena CoP meningkat secara monoton dengan T_{sup} (lihat Gambar 1.7). Namun, menurut Persamaan. 1.5, T_{sup} bisa paling sama dengan suhu masuk maksimum server.



Gambar 1.7. Contoh grafik CoP

Karena pembangkitan panas server secara langsung tergantung pada beban kerja mereka, penjadwalan pekerjaan sadar termal dapat digunakan untuk meminimalkan resirkulasi panas di antara server, yang memungkinkan CRAC beroperasi pada suhu yang lebih tinggi, dan karenanya efisiensi.

Solusi Berbasis Heuristik dan GA untuk Penjadwalan Pekerjaan Sadar Termal di Pusat Data

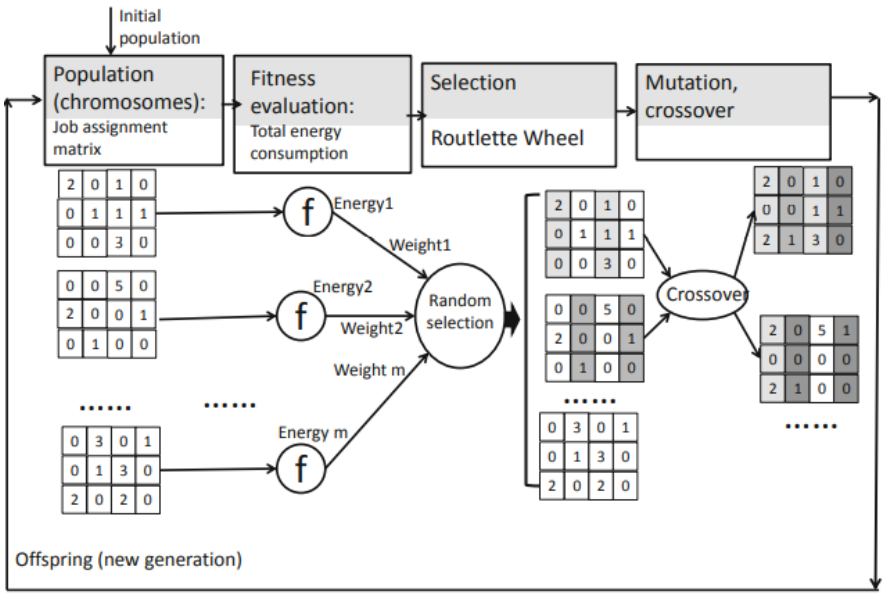
Solusi heuristik untuk penjadwalan pekerjaan sadar termal diusulkan untuk meminimalkan resirkulasi panas total (yaitu, MinHR, dan LRH) dan untuk mendistribusikan beban kerja ke server berbanding terbalik dengan suhu masuk server. Penelitian sebelumnya memanfaatkan GA untuk penjadwalan pekerjaan sadar termal di pusat data HPC dan mengusulkan masing-masing XInt-GA, dan SCINT. Melalui studi simulasi jejak pekerjaan yang realistis, penulis menemukan bahwa solusi berbasis GA (yaitu, XInt-GA, dan SCINT) memiliki manfaat hemat energi yang jauh lebih tinggi daripada solusi heuristik yang disebutkan di atas tetapi juga membutuhkan waktu lebih lama untuk menyelesaikannya (beberapa jam dibandingkan dengan sepersekian detik). Gambaran singkat XInt-GA dan SCINT diberikan di bawah ini.

XInt-GA adalah solusi berbasis GA untuk penjadwalan pekerjaan sadar termal spatio di pusat data virtual dan homogen di mana semua server mampu menjalankan semua pekerjaan dengan kecepatan yang sama. Dalam model pusat data seperti itu, meminimalkan resirkulasi panas adalah masalah optimisasi min-maks kombinatorial, yaitu NP-complete'. Pertimbangkan vektor penugasan pekerjaan c di mana setiap elemen, ci, menentukan berapa banyak prosesor dari server i ditugaskan untuk pekerjaan itu. Masalah penugasan pekerjaan sadar termal dapat diringkas sebagai berikut:

$$\begin{aligned} &\min \max_i && T_{in,i} \\ &s.t.: && C_{tot} - \sum_{i=1}^{N} c_i \\ & && \boldsymbol{T}_{in} = T_{\boldsymbol{red}} - \boldsymbol{D}(\boldsymbol{a} \odot \boldsymbol{c}) + D\boldsymbol{b} \\ & && c_i \leq m_{\boldsymbol{i}} \end{aligned} \tag{1.6}$$

DImana C_{tot} menunjukkan jumlah total prosesor yang dibutuhkan pekerjaan, $\boldsymbol{b}$ menunjukkan vektor daya idle server, dan $\boldsymbol{a}$ menunjukkan vektor konsumsi daya server untuk masing-masing prosesornya. Di IXint-GA, set awal dari solusi yang layak adalah semua vektor penugasan pekerjaan yang memenuhi persyaratan kinerja pekerjaan. Sebuah gen adalah setiap elemen dari vektor penugasan pekerjaan, dan fungsi kebugaran adalah suhu masuk puncak yang dihasilkan dari penugasan pekerjaan tersebut. Generasi baru dipilih dengan proses pemilihan Roulette Wheel berbasis probabilitas yaitu a. *** definisi atau kutipan untuk Routlette Wheel.*

Kinerja algoritma XInt-GA terhadap dua algoritma lain yang disebut XInt-SQP dan MinHR menggunakan pusat data skala kecil yang disimulasikan. XInt-SQP menggunakan pemrograman kuadratik sekuensial untuk menyelesaikan formulasi minimax Persamaan. 1.6 dalam domain bilangan real. Yang berarti bahwa vektor penugasan pekerjaan, c, harus dilonggarkan ke variabel kontinu, yaitu, lebih dari satu pekerjaan ditugaskan ke prosesor. Solusi selanjutnya didiskritisasi untuk menemukan solusi bilangan bulat dekat dan layak. MinHR adalah solusi heuristik yang meminimalkan resirkulasi panas total alih-alih meminimalkan sirkulasi ulang panas maksimum. Hasil menunjukkan bahwa algoritme XInt-GA mencapai penghematan biaya maksimum (hingga 30% tergantung pada beban pusat data) di antara semua solusi lainnya. Diskusikan XInt-SQP.



Gambar 1.8. Diagram untuk algoritma XInt-GA

Penelitian yang dilakukan Mukharejee memperkenalkan penjadwalan pekerjaan Spatio-Temporal sadar termal, sebuah alternatif untuk yang diperkenalkan di atas, yang

memutuskan baik waktu mulai pelaksanaan pekerjaan dan tugas ke server tugas. Dengan memanfaatkan kelonggaran dalam jadwal pelaksanaan pusat data, pekerjaan dapat ditunda selama tenggat waktunya memungkinkan untuk membuat jadwal yang lebih lancar di mana peralatan yang paling hemat energi digunakan sebanyak mungkin. Hal ini dapat digabungkan dengan pemahaman tentang heterogenitas peralatan untuk menjadwalkan pekerjaan pada peralatan yang paling hemat energi. Secara intuitif, peralatan tertentu lebih hemat energi per komputasi daripada peralatan lain di lingkungan yang heterogen. Dengan menggunakan peralatan yang paling efisien, sebanyak mungkin energi dihemat baik dengan menghabiskan lebih sedikit per komputasi maupun dengan pendinginan lebih sedikit karena semua energi yang dihabiskan muncul sebagai panas dan dengan demikian perlu didinginkan. Menggabungkan faktor-faktor ini secara sinergis memberikan bentuk penjadwalan dengan efek penghematan energi super-linear.

Pertimbangkan sistem slot waktu diskrit untuk penjadwalan. Dalam sistem seperti itu, penjadwalan yang optimal hanya mungkin dicapai bila kita memiliki pengetahuan yang sempurna tentang runtime pekerjaan. Dalam praktiknya, karena runtime pekerjaan sebenarnya hanya diketahui saat pekerjaan selesai, jadwal yang optimal tidak mungkin dicapai. Misalnya, jika dalam jadwal pekerjaan awal kami, peralatan yang paling efisien di pusat data digunakan sepenuhnya dan kemudian pekerjaan pada peralatan itu selesai sebelum yang kami harapkan, waktu kendur baru pada peralatan ini mungkin tidak dapat diisi secara optimal sambil mempertahankan SLA. Seandainya kami memiliki perkiraan waktu kerja pekerjaan yang sempurna sebelumnya, kami dapat membuat jadwal yang berbeda di mana pekerjaan yang dijalankan secara paralel mungkin dapat ditunda untuk mengisi waktu luang ini dengan cara yang lebih efisien.

Secara formal masalah ini dapat didefinisikan untuk meminimalkan total konsumsi energi sebagai berikut:

min:

$$\sum_{t=1}^{T}\left(\sum_{i=1}^{N}\left(\boldsymbol{a}\odot\boldsymbol{c_{i,t}}\right)+\boldsymbol{b}\right)\ +\ \frac{\sum_{i=1}^{N}\left(\boldsymbol{a}\odot\boldsymbol{c_{i,t}}\right)+\boldsymbol{b}\right)}{CoP\left(T_{r}ed-max_{i}D\left(\boldsymbol{a}\odot\boldsymbol{c_{i,t}}\right)+\boldsymbol{b}\right)}\Bigg)\tau$$

s.t.:

$$\sum_{k=1} k\ C_{i,k,t}=C_{i,t}\leq\ m_{i}\forall i\leq N,dan\ ,t\leq T\ [capacity\ constraint]$$

$$\sum_{i=1}^{N} C_{i,k,t}\geq\ C_{tot,}k\ [perfomance\ constraint]$$

$$[jobs'deadline\ constraint]$$

$$[jobs'arrival\ time\ constraint]$$

(1.7)

dimana k menunjukkan indeks pekerjaan, K menunjukkan jumlah pekerjaan, dan τ menunjukkan interval waktu untuk pengambilan keputusan. Istilah pertama dan kedua dalam fungsi tujuan masing-masing menunjukkan energi komputasi dan energi pendinginan. Kendala

tenggat waktu dan waktu kedatangan dihilangkan di atas demi singkatnya. Karena masalah ini adalah NP-hard, solusi optimal offline tidak dapat diselesaikan dalam waktu polinomial. Oleh karena itu, solusi berbasis GA, Scheduling untuk meminimalkan thermal *cross-INTERference* (SCINT), diusulkan untuk menyelesaikan masalah secara heuristik secara offline. Dengan cara ini, populasi awal diunggulkan dari penjadwal pekerjaan dasar yang layak (misalnya, FIFO). Fungsi kebugaran adalah konsumsi energi total. *Crossover* (kawin) dan mutasi dilakukan untuk menghasilkan solusi baru dan mengacak gen dalam individu masing-masing. Generasi baru dipilih dari kumpulan solusi saat ini menggunakan algoritma pemilihan Roulette Wheel.

Penelitan dari Mukharejee et al memberikan hasil melalui pusat data simulasi pusat data HPCI ASU yang matriks resirkulasi panasnya diperoleh dengan perangkat lunak simulasi CFD. Performa SCINT dibandingkan dengan solusi heuristik lainnya termasuk EDF-LRH online (*Earliest Deadline First Least Recirculated Heat)* menggunakan log penyerahan pekerjaan dari pusat data di atas. EDF-LRH menggunakan server heuristik dan peringkat secara statis dengan menggunakan metrik LRH untuk meminimalkan resirkulasi panas total. LRH mengurutkan server berdasarkan kontribusinya pada resirkulasi panas secara statis saat diasumsikan digunakan sepenuhnya. Pekerjaan ditugaskan ke server yang tersedia yang peringkat LRH-nya paling rendah. Hasil menunjukkan bahwa SCINT menghemat hingga 40% lebih banyak energi daripada penjadwalan pekerjaan naif yang tidak sadar energi (yaitu, FCFS) dan 23% lebih banyak energi daripada EDF-LRH. Namun, SCINT adalah algoritme offline yang memerlukan beberapa jam untuk diproses dibandingkan dengan beberapa milidetik yang diperlukan oleh EDF-LRH.

Rangkuman Hasil

Penelitian dari Mukharejee et al dengan jelas menunjukkan tradeoff biaya-kinerja untuk solusi berbasis GA dibandingkan dengan solusi heuristik. Penelitian ini juga menunjukkan aplikasi GA untuk mengevaluasi solusi heuristik. Dengan kata lain, karena solusi optimal eksak karena nonlinieritas fungsi tujuan, variabel diskrit, dan kekerasan NP dari masalah tidak dapat ditemukan tanpa pencarian brute-force, solusi berbasis GA dimanfaatkan untuk mengevaluasi solusi heuristik. Solusi berbasis GA menghasilkan manfaat penghematan energi yang signifikan dalam biaya durasi pengoperasian yang tinggi. Ini adalah motivasi untuk penelitian masa depan di bidang ini untuk menemukan solusi berbasis GA yang cepat

Survei Penjadwalan Komunikasi Sadar Termal di Jaringan Biosensor Implan

Untuk beberapa jenis WSN seperti BSN, keamanan termal sangat penting. Di BSN, komunikasi dalam jaringan biosensor yang ditanamkan yang digunakan untuk pemantauan kesehatan dapat diatur ke dalam kelompok di mana sebagian besar komunikasi berlangsung. Namun, cluster yang berjalan dalam waktu lama dapat menghasilkan panas yang cukup untuk merusak jaringan di sekitarnya. Penelitian Tang mengusulkan satu set solusi heuristik dan solusi berbasis GA untuk memastikan keamanan termal jaringan di mana jaringan biosensor ditanamkan. Penulis mengusulkan rotasi pemimpin klaster untuk mencegah potensi panas berlebih pada jaringan termal. Efek termal dari biosensor yang ditanamkan dimodelkan dengan menghitung Tingkat Penyerapan Spesifik (SAR) dan mendiskritisasi peningkatan suhu

terhadap ruang dan waktu dengan Domain Waktu Perbedaan Terbatas (FDTD). Model menangkap interferensi panas di antara node. Singkatnya, masalah didefinisikan sebagai: diberikan volume kontrol, lokasi yang diketahui dari sensor yang ditanamkan, dan properti terkait, bagaimana cara menjadwalkan urutan rotasi untuk meminimalkan kenaikan suhu. Untuk memodelkan dinamika suhu biosensor, makalah menganggap bahwa peningkatan suhu jaringan dipengaruhi oleh tiga sumber utama sebagai berikut:

- Pemanasan yang disebabkan oleh RF powering. Penulis menganggap node sensor diisi ulang oleh sumber daya RF eksternal dan proses pengisian ulang ini akan memanaskan jaringan di sekitarnya.
- Radiasi dari antena node sensor. Jaringan yang mengelilingi antena sebagian menyerap energi transmisi antena.
- Disipasi daya oleh sirkuit node sensor. Saat perhitungan dilakukan, panas hilang.

Mengingat sumber panas di atas, penulis memodelkan suhu di atas ruang dan waktu. FDTD digunakan untuk mendiskritisasi persamaan variasi suhu terhadap ruang (menggunakan sel 2-D kecil) dan waktu, sedemikian rupa sehingga suhu setiap sel merupakan fungsi dari suhu sebelumnya serta suhu sebelumnya dari tetangganya, yaitu suhu di titik (*i,j*) pada waktu *m* + 1 merupakan fungsi suhu di titik (*i,j*), (*i* + 1*, j*), (*i, j* + 1), (*i* - 1, j), dan (*i, j* - 1) pada waktu m.

Model berbasis FDTD di atas mahal secara komputasi. Untuk mengatasi hal ini, penulis mengusulkan model yang disederhanakan yang memperkirakan suhu maksimum yang mungkin untuk setiap rotasi pemimpin klaster. Kenaikan ini adalah metrik Potensi Kenaikan Suhu (TIP). Dalam model ini tingkat kenaikan suhu dari sebuah node adalah fungsi dari jarak euclidean dari pemimpin saat ini dan waktu sejak sebelumnya menjadi pemimpin. Hasil numerik FDTD digunakan untuk melatih model yang disederhanakan. Kemudian metrik TIP digunakan untuk menghitung penjumlahan pengaruh kepemimpinan node i pada semua node lainnya. Mengingat metrik TIP, menemukan rotasi pemimpin yang meminimalkan suhu maksimum masih merupakan masalah NP-hard yang serupa dengan masalah Travelling Sales Man (TSP). Solusi waktu polinomial untuk masalah ini tidak ada. Oleh karena itu, penulis mengusulkan metode berbasis GA untuk mencari rotasi mendekati optimal sebagai berikut:

Kromosom setiap individu adalah urutan rotasi. Metrik TIP setiap individu digunakan sebagai ukuran kebugaran. Oleh karena itu, rotasi yang lebih aman secara termal lebih mungkin bertahan dalam proses evolusi GA. Urutan awal dipilih secara acak dari polling solusi yang layak. Crossover acak digunakan di mana individu dipilih secara acak. Kromosom ditukar untuk menghasilkan dua individu baru untuk generasi berikutnya. Untuk proses mutasi, individu dengan probabilitas rendah (nilai fitness rendah) dipilih dan dua gen dalam rotasi dipilih secara acak dan ditukar untuk membuat rotasi baru. Populasi baru terdiri dari pemilihan acak berbobot dari individu sebelumnya dan anak yang baru dihasilkan dimana individu dengan nilai fitness yang lebih tinggi lebih mungkin untuk bertahan hidup. Penulis tidak memberikan evaluasi algoritma GA dibandingkan dengan heuristik manapun.

Survei Pemanenan Energi dan Manajemen Biaya di Pusat Data

Harga listrik di pusat data bervariasi dari waktu ke waktu dan lokasi. Juga banyak pusat data memanfaatkan sumber energi terbarukan bawaan untuk berkontribusi dalam komputasi hijau. Baru-baru ini komunitas riset mengusulkan untuk memanfaatkan variasi spatio-

temporal harga listrik yang disebutkan di atas, dan jenis pasokan energi (yaitu, hijau/coklat) berdasarkan beban kerja dan manajemen server di seluruh pusat data untuk pekerjaan jenis web, serta pekerjaan tipe batch. Namun karena nonlinieritas model biaya energi yang bergantung pada daya idle server dan daya dinamis, overhead migrasi pekerjaan dan persyaratan penundaan pekerjaan yang mencegahnya berjalan di pusat data tertentu, bentuk online dari masalah tersebut adalah pemrograman bilangan bulat, yang umumnya NP-hard. Untuk itu literatur mengusulkan heuristik dan meta-heuristik untuk menyelesaikannya.

Penelitian dari Qureshi menggunakan heuristik untuk mengukur keuntungan ekonomi potensial dengan mempertimbangkan harga listrik di lokasi perhitungan. Penulis memperkirakan masalah manajemen beban kerja pusat data dengan model aliran biaya minimum. Penulis selanjutnya memperbaiki skema mereka dengan mengusulkan optimasi bersama kontrol daya dan harga listrik. Mereka mengembangkan masalah pengoptimalan nonlinier yang mengontrol penskalaan tegangan server serta jumlah server aktif dan menyelesaikannya dengan metode dekomposisi benders umum. Mengembangkan solusi serakah online untuk mengatasi overhead migrasi pekerjaan dalam masalah tersebut. Metode anil simulasi untuk memecahkan masalah server pusat data dan manajemen beban kerja.

Rangkuman Hasil

Literatur mengidentifikasi peluang untuk meningkatkan efisiensi energi pusat data melalui manajemen biaya energi di seluruh pusat data. Mereka memodelkan masalah dan melaporkan penghematan biaya yang signifikan dari masalah ini dengan menggunakan data realistis. Namun solusi yang efisien untuk masalah ini belum dikembangkan. Solusi heuristik dan meta-heuristik yang diusulkan belum dibandingkan dalam hal kinerja penghematan biaya dan efisiensi waktu/kompleksitas komputasi. EA akan berpotensi membantu kinerja dari solusi masalah yang dapat dianggap sebagai pekerjaan masa depan.

1.5 KESIMPULAN

Bab ini memberikan gambaran umum tentang masalah komputasi hijau di DCPS yang diselesaikan dengan menggunakan teknik evolusioner. taksonomi penelitian di bidang ini diberikan pada Tabel. 1.1. Solusi komputasi ramah lingkungan untuk DCPS dapat dicapai melalui penjadwalan beban kerja sadar energi dan termal antar node. Namun, masalah tersebut berubah menjadi NP-hard, di mana solusi optimal tidak dapat dihitung dengan cara yang efisien waktu. Oleh karena itu, literatur mengembangkan solusi heuristik dan meta-heuristik menggunakan EA. Secara umum, solusi berbasis EA terbukti memiliki kinerja yang lebih tinggi daripada rekan heuristiknya, tetapi mungkin membutuhkan waktu lebih lama untuk menyelesaikannya.

Tabel 1.1. Taksonomi penelitian tentang penerapan teknik evolusioner pada komputasi hijau di DCPS

Domain	Sistem Terdistribusi	Masalah	Skema
ruang maya	Pusat Data	penjadwalan beban kerja	meta-heuristik (yaitu, GA, ACO)

	Pusat Data	penjadwalan beban kerja	heuristis
	WSN	masalah perutean	meta-heuristik (GA, ED, ACO)
	WSN	masalah perutean	heuristis
fisik maya	Pusat Data	jadwal beban kerja sadar termal.	meta-heuristik (GA)
	Pusat Data	jadwal beban kerja sadar termal.	heuristis
	WSN	jadwal beban kerja sadar termal. dan kom.	meta-heuristik (GA)
	Pusat Data	manajemen biaya energi di seluruh pusat data	meta-heuristik (anil simulasi)
	Pusat Data	manajemen biaya energi di seluruh pusat data	heuristis

Studi saat ini menunjukkan bahwa komunitas riset berhasil memanfaatkan solusi berbasis EA untuk memodelkan distribusi beban kerja sadar energi dan termal di DCPS. Namun, penelitian ini juga menunjukkan bahwa banyak pekerjaan yang tersisa untuk pekerjaan masa depan di bidang ini. Jaminan teoretis biasanya tidak ada untuk kinerja solusi berbasis EA.

Oleh karena itu, untuk memastikan kinerja skema EA dalam praktiknya, diperlukan studi simulasi/eksperimen yang komprehensif yang menguji kinerja solusi pada berbagai jenis data input. Sebagian besar solusi yang diusulkan tidak dievaluasi dengan benar. Dengan kata lain, banyak literatur memberikan hasil sesuai dengan input data yang terbatas. Banyak dari mereka kekurangan dari studi perbandingan komprehensif yang secara tepat menunjukkan seberapa jauh kinerja yang lebih baik yang dicapai oleh solusi berbasis EA untuk nilai biaya apa dibandingkan dengan solusi heuristik atau solusi EA lainnya. Berbagai jenis solusi EA digunakan untuk masalah komputasi hijau di DCPS, tetapi tidak ada studi yang membandingkan kinerjanya (dalam hal optimalitas solusi, dan efisiensi waktu komputasi skema sehubungan dengan masing-masing solusi). lain Singkatnya, studi yang ditinjau menunjukkan bahwa masalah komputasi hijau dapat berhasil dimodelkan dengan pendekatan EA, tetapi hasilnya jauh dari pendekatan pemanfaatan yang tepat dari peluang yang diberikan EA.Selanjutnya, banyak penelitian menunjukkan bahwa kinerja Solusi EA dapat ditingkatkan secara signifikan dengan menggabungkan heuristik khusus aplikasi dalam proses pengambilan keputusan EA.Studi di masa depan harus menekankan pada skema tersebut untuk mencapai solusi berbasis EA yang lebih efisien waktu dan kinerja lebih tinggi daripada yang sudah ada.

Baru-baru ini pemanenan energi dan manajemen biaya energi di DCPS diusulkan oleh beberapa literatur. Masalah-masalah ini sangat kompleks untuk mengatasi ketidakpastian yang terkait dengan sumber energi hijau, dan fungsi utilitas nonlinier untuk menyesuaikan konsumsi energi dengan profil biaya rendah/energi hijau yang terbatas sambil mempertahankan kualitas beban kerja layanan. Penerapan EA untuk masalah tersebut adalah area penelitian terbuka lainnya.

BAB 2
PENYEDIAAN LAYANAN HPC
MELALUI VIRTUALISASI LAYANAN WEB

Dalam bab ini kami menyajikan infrastruktur pendukung untuk Organisasi Virtual yang memungkinkan pengiriman dan distribusi produk layanan TI yang kompleks dengan cara yang terkendali namun terbuka. Penyediaan layanan di bawah paradigma seperti itu biasanya membutuhkan lingkungan khusus dan basis pengetahuan serta alat. Secara khusus, di lingkungan HPC (*High Performance Computing*), pengetahuan mendetail tentang sistem yang tersedia dan perilaku serta performanya sangat penting karena menawarkan cara untuk menghindari penurunan performa dan peningkatan biaya terkait. Selain itu, lingkungan HPC dicirikan oleh permintaan energi yang sangat tinggi. Namun, sebagian besar pengguna infrastruktur semacam itu tidak menyadari teknologi yang mereka gunakan, memaksakan keterlibatan para ahli untuk menghindari penggunaan sumber daya yang tidak optimal, dan dengan demikian membuang-buang waktu, tenaga dan uang. Kami menunjukkan contoh produk TI yang begitu kompleks dengan menjelaskan proses sampel di bidang komputasi medis - simulasi tulang cancellous - dan mengilustrasikan bagaimana produk yang begitu kompleks dapat disediakan dengan cara yang mudah digunakan dan hemat energi. mode melalui infrastruktur virtualisasi layanan.

2.1 PENDAHULUAN

Selain penggunaan Internet sebagai sumber informasi, penggunaan Web sebagai platform interaksi juga semakin populer untuk penyediaan berbagai jenis layanan, seperti layanan rantai pasokan, layanan telekomunikasi, layanan e-commerce, atau lainnya. - layanan penyediaan energi, untuk menyebutkan beberapa saja. Selain itu, dengan kecenderungan komputasi awan, Internet telah mendapatkan minat tambahan dari para pemangku kepentingan sebagai platform komunikasi dan interoperabilitas yang memungkinkan penyediaan hampir semua jenis sumber daya, berkat ketersediaan data dan standar komunikasi. Dengan mempertimbangkan infrastruktur ini yang memungkinkan pengintegrasian berbagai layanan dan sumber daya, yang mungkin didistribusikan ke seluruh dunia, konsep Organisasi Virtual (VO) telah digunakan secara luas sejak awal abad ke-21. Model VO, meramalkan untuk menangani sumber daya virtual, biasanya tersedia melalui Internet, untuk menyediakan produk agregat dan kompleks yang terdiri dari berbagai layanan orkestra. Pendekatan ini telah dipertimbangkan dalam berbagai macam proyek penelitian. Secara khusus, kemampuan yang diberikan oleh paradigma VO untuk menyediakan lingkungan yang dinamis dan dapat menyembuhkan diri sendiri, mampu bereaksi secara mandiri terhadap kejadian tak terduga, seperti kinerja yang buruk atau kegagalan penyedia layanan subkontrak, telah menjadi salah satu pendorongnya. proyek Penelitian. Namun, pengelolaan lingkungan yang dinamis tersebut masih dianggap sebagai salah satu tugas yang

paling menantang, karena kesulitan teknis yang berkaitan dengan kemampuan mereka untuk sepenuhnya memanfaatkan kemampuan beradaptasi dan kesadaran diri.

Tantangan-tantangan ini terkait misalnya dengan berbagai aspek yang harus dipertimbangkan ketika penyedia layanan digantikan oleh yang lain, misalnya karena kegagalan layanan. Menemukan layanan pengganti yang kompatibel, menggantinya di bawah masalah kontrak yang benar, dan menghubungkannya dengan benar ke jaringan layanan adalah beberapa tugas yang menantang ini. Ini bahkan lebih penting dalam tugas-tugas yang terkait dengan manajemen sumber daya internal dalam suatu organisasi. Dalam arah tersebut, Perencanaan Sumber Daya Perusahaan (ERP) dan Integrasi Aplikasi Perusahaan (EAI) menjadi semakin penting untuk penggunaan sumber daya internal yang efisien, sambil menghadapi tantangan serupa saat mengoperasikan VO yang terdiri dari layanan eksternal tunggal.

Solusi konseptual dan praktis untuk sebagian besar masalah di atas telah ditawarkan oleh paradigma Service Oriented Architecture (SOA), yang memetakan ide-ide mapan dan terbukti menyediakan antarmuka standar - abstrak dari realisasi fungsi yang mendasari - dari dunia teknik elektro ke dunia rekayasa perangkat lunak.

Untuk menempatkan pendekatan seperti itu di tempat kerja, teknologi layanan web telah dilihat sebagai pengaktif untuk SOA nyata dalam lingkungan terdistribusi berdasarkan teknologi web. Namun, pendekatan ini hanya berhasil dalam cara yang terbatas, karena, karena pertumbuhan berkelanjutan di bidang standar layanan web, tujuan awal, yaitu memisahkan definisi antarmuka dari implementasi layanan, yang memungkinkan konsumen layanan untuk meminta metode antarmuka, belum sepenuhnya tercapai karena kesulitan teknis, terikat untuk mengontrol sumber daya terdistribusi, masalah keamanan dan ketersediaan, dan kesulitan untuk membangun repositori layanan web (dalam gaya UDDI) yang dapat digunakan di dunia - lingkungan komersial yang luas.

Penggunaan VO seperti itu berdasarkan layanan web malah menjadi lebih diterima dan digunakan dalam lingkungan ilmiah, di mana pembagian sumber daya digunakan misalnya untuk melakukan eksperimen besar yang membutuhkan sumber daya khusus (misalnya, sumber daya komputasi kinerja tinggi atau basis data sampel dan data yang besar). Sejauh ini, hanya orang-orang yang berorientasi teknis - dan ilmiah - tertarik untuk menyadari dan menggunakan proses perangkat lunak berdasarkan layanan web dalam gaya VO, meskipun model cloud berdasarkan kumpulan layanan web yang akan dipanggil dengan gaya VO -datang menarik bagi dunia komersial dan perusahaan juga.

Secara khusus, pengalaman TI pengguna yang berbeda menunjukkan bahwa lingkungan yang tersedia saat ini tidak memuaskan persyaratan dan kebutuhan konkret mereka. Kami menunjukkan contoh produk TI yang begitu kompleks dengan menjelaskan proses sampel di bidang komputasi medis - simulasi tulang cancellous - dan mengilustrasikan bagaimana produk kompleks semacam itu dapat disediakan dengan cara yang mudah digunakan melalui infrastruktur virtualisasi layanan. sambil menganalisis jejak konsumsi energi yang sesuai.

Simulasi semacam ini adalah proses yang sangat kompleks, membutuhkan sejumlah besar pengetahuan terperinci tentang infrastruktur komputasi dan penyiapan infrastruktur. Melalui contoh ini, kami menunjukkan bagaimana proses yang kompleks dapat disediakan sebagai layanan TI untuk disediakan dan dikonsumsi sesuai dengan paradigma VO layanan web, di mana virtualisasi infrastruktur dan lingkungan yang mendasarinya memungkinkan pemanggilan proses simulasi di dalamnya. bahasa alur kerja yang umum, seperti mis. BPEL.

Kami menganalisis konsumsi energi lingkungan dan mengarahkan penyebaran dan pelaksanaan simulasi yang sesuai. Secara khusus, hal ini memungkinkan pengguna dengan sedikit keterampilan TI untuk menggunakan layanan yang kompleks tersebut dengan beban terbatas yang diperlukan untuk memperoleh pengetahuan tentang lingkungan yang digunakan. Untuk mencapai alur kerja proses hemat energi dalam lingkungan tervirtualisasi, kami menjelaskan pendekatan kami tentang penyediaan dan alokasi hemat energi sumber daya HPC tervirtualisasi. Secara khusus, kami menjelaskan bagaimana hal ini dicapai dengan mempertimbangkan kendala tentang persyaratan Kualitas Layanan (QoS), melalui konsolidasi sumber daya dan teknik migrasi, atau dengan menyesuaikan status node fisik dengan mengubah, misalnya, P-State CPU.

Oleh karena itu, kami menerapkan Loop Kontrol Global dari infrastruktur proyek GAMES ke lingkungan virtualisasi layanan kami, untuk memungkinkan pengarahan penyebaran pekerjaan yang transparan dengan mempertimbangkan efisiensi energi dari seluruh proses.

2.2 ALUR KERJA ILMIAH DENGAN DESKRIPSI ALUR KERJA UMUM

Pada bagian ini, kami menganalisis persyaratan untuk alur kerja ilmiah, dan menyoroti dalam mode mana standar "de-facto" untuk alur kerja di eBusiness, yaitu bahasa alur kerja BPEL (Business Process Execution Language), dapat digunakan untuk memberlakukan dan menyebarkan proses ilmiah HPC.

Persyaratan Lingkungan Alur Kerja Ilmiah

Para ilmuwan biasanya berspesialisasi dalam domain aplikasi mereka sementara tidak (dan seringkali tidak tertarik untuk) mengetahui infrastruktur TI dan layanan yang mendasari yang digunakan untuk eksperimen dan simulasi mereka. Namun, karena beberapa lingkungan aplikasi saat ini memerlukan pengetahuan tertentu tentang infrastruktur TI dan lingkungan perangkat lunak, khususnya di lingkungan HPC dan simulasi, untuk mengatur percobaan, untuk memilih sumber daya yang berguna (data dan program) dari platform atau untuk memilih mitra tepercaya yang bekerja sama selama percobaan, para ilmuwan sering kali dipaksa untuk berkenalan.

Pengetahuan dasar ini biasanya tidak cukup untuk melaksanakan dan menindaklanjuti proses simulasi, atau untuk menghadapi masalah khusus yang terkait dengan infrastruktur TI, mis. jika lingkungan TI tidak berperilaku seperti yang diharapkan (misalnya, ketika kegagalan atau pengecualian muncul). Ini mewajibkan para ilmuwan untuk meminta dukungan dari pakar TI, yang biasanya menyebabkan penundaan dan ketidaknyamanan bagi para ilmuwan dan pakar TI. Oleh karena itu, disarankan untuk memisahkan infrastruktur TI dari logika aplikasi, sehingga mencerminkan pemisahan antara pengetahuan ilmiah tentang aplikasi dari

pengetahuan TI tentang penyediaan layanan. Secara khusus, decoupling ini memungkinkan manajemen profesional yang lebih baik dari lingkungan infrastruktur TI, sementara para ilmuwan dapat berkonsentrasi pada aspek penelitian yang terkait dengan domain ilmiah yang dipelajari.

Namun demikian, para ilmuwan masih tertarik untuk mengontrol eksekusi alur kerja, karena, selama simulasi, khususnya jika terjadi fenomena yang tidak terduga, menarik untuk menunda eksekusi, atau mengubah beberapa parameter dan kemudian melanjutkan simulasi. Selain itu, jika terjadi fenomena yang tidak terduga, menarik juga untuk mengekstraksi hasil perhitungan antara dan sementara.

Selain itu, selama pelaksanaan proses simulasi, penyempurnaan struktur alur kerja mungkin diperlukan, karena hasil simulasi antara. Modifikasi/evolusi mode eksekusi proses ini dapat disebabkan oleh perilaku tak terduga dari komponen atau elemen tertentu yang terkait dengan domain aplikasi, bukan infrastruktur TI.

Isu lain yang harus dipertimbangkan dalam konteks lingkungan simulasi ilmiah adalah bahwa proses ini seringkali merupakan proses yang berjalan lama. Selain itu, infrastruktur pekerjaan komputasi HPC tersebut diganti secara berkala karena perkembangan dan peningkatan yang cepat di sektor perangkat keras TI. Pengembangan ini, yang memungkinkan daya komputasi berkinerja lebih tinggi dengan konsumsi energi lebih sedikit, biasanya memerlukan penggantian arsitektur TI setelah kira-kira 5 tahun, karena sistem perlu menyesuaikan persyaratan penghematan energinya, misalnya, sepele, karena kenaikan harga tenaga listrik yang terus menerus dan meningkatnya minat masyarakat dalam penghematan energi dan dampak lingkungan. Oleh karena itu, adaptasi infrastruktur TI adalah evolusi umum, menyiratkan juga evolusi lingkungan simulasi karena perangkat keras yang berubah (misalnya, infrastruktur TI baru yang memungkinkan kinerja yang lebih baik untuk pekerjaan tertentu dengan efisiensi energi yang lebih baik memerlukan adaptasi dalam penelitian ilmiah). struktur proses menghindari siklus yang tidak perlu atau permintaan layanan perangkat lunak yang lebih berkinerja). Di lingkungan HPC, sumber daya yang heterogen digunakan untuk menghadapi kebutuhan yang berbeda dari berbagai pekerjaan komputasi, menyebabkan biaya tambahan bagi para ilmuwan untuk menentukan konfigurasi optimal dari pekerjaan ilmiah tertentu.

Gagasan mengembangkan alur kerja ilmiah bukanlah hal baru. Sebenarnya, berbagai Sistem Manajemen Alur Kerja (WFMS) tersedia yang mendukung perubahan struktur proses di bawah berbagai kendala dan pengecualian. Namun, pendekatan ini biasanya terspesialisasi dan terbatas pada domain tertentu, menyebabkan masalah jika terjadi kolaborasi dalam domain antar-disiplin.

Keunikan lain dari alur kerja ilmiah adalah bahwa hasilnya sering dicapai dengan beberapa eksekusi proses simulasi dengan berbagai parameter (studi parameter) dengan cara *trial-and-error*. Model komputasi ini disebabkan oleh fakta bahwa untuk sebagian besar masalah dunia nyata sulit untuk menemukan model matematika untuk menggambarkan seluruh pernyataan masalah. Oleh karena itu, simulasi, khususnya studi parameter, dilakukan untuk menemukan solusi optimal dengan mencoba berbagai pengaturan yang berbeda.

Setelah hasil tercapai, penting juga untuk memungkinkan reproduktifitas hasil yang dicapai, sehingga memungkinkan ketertelusuran hasil.

Menerapkan Bahasa Deskripsi Alur Kerja Umum ke Domain Komputasi Ilmiah

Proses simulasi mengikuti aliran yang terdefinisi dengan baik, yang dapat dijelaskan menggunakan bahasa alur kerja yang umum. Secara khusus, standar de-facto untuk alur kerja berbasis layanan web, yaitu BPEL, secara umum dapat digunakan untuk alur kerja ilmiah. Target utama lingkungan alur kerja umum adalah otomatisasi proses yang terdefinisi dengan baik, yang memetakan secara teoritis ke prinsip studi parameter. Studi-studi ini tercermin dari pelaksanaan proses yang sama dengan parameter input yang bervariasi, hingga tujuan tertentu tercapai.

Biasanya, dalam proses ilmiah sejumlah besar data harus diproses, sehingga penanganan yang sesuai adalah masalah mendasar. Namun, karena BPEL terutama berkonsentrasi pada aliran kontrol tetapi tidak pada aliran data, BPEL menunjukkan beberapa celah dalam hal ini. Yaitu, BPEL tidak memiliki dukungan yang sesuai untuk penanganan kumpulan data besar pada tingkat alur kerja, perpipaan, berbagi data antara contoh yang berbeda dan berpotensi didistribusikan, serta pandangan abstraksi yang berbeda pada kumpulan data yang sesuai. Selain itu, kumpulan data ini seringkali bersifat heterogen, jadi, untuk memungkinkan penggunaan mesin alur kerja dalam konteks ini, pengiriman tautan secara eksplisit antara kumpulan data ini, serta pemetaan potensial dari berbagai tipe data, harus diintegrasikan secara transparan.

Dalam lingkungan seperti itu, ilmuwan biasanya bertindak sebagai pemodel, pengguna, administrator, dan analis alur kerja. Seringkali, pengembangan alur kerja ilmiah biasanya dilakukan dengan cara "coba-coba", sementara alur kerja klasik ini menjelaskan alur kontrol aplikasi/proses yang tidak mempertimbangkan aliran data yang sesuai. Kesenjangan utama di sini adalah bahwa bahasa alur kerja umum tidak menyediakan mekanisme yang masuk akal untuk menangani kumpulan data besar, yang harus diproses selama pelaksanaan alur kerja ilmiah.

Teknologi alur kerja yang disebutkan sudah ada di domain eBusiness, di mana fokus utamanya bergantung pada deskripsi proses yang terkenal. Karena proses yang terstruktur dengan baik dalam lingkungan simulasi, memungkinkan adaptasi dari teknologi alur kerja umum yang sudah mapan ini dalam konteks ini, komunitas ilmiah semakin tertarik dengan WfMSs.

WfMS ini, khususnya dalam kombinasi dengan lingkungan berbasis SOA, memungkinkan pelaksanaan eksperimen dalam lingkungan yang sangat heterogen. Khususnya di lingkungan HPC ini adalah masalah yang sangat menantang, karena di dalam lingkungan ini biasanya melibatkan beberapa mesin yang berbeda untuk jenis pekerjaan yang berbeda. Hal ini disebabkan oleh fakta bahwa simulasi spesifik cenderung berjalan pada platform perangkat keras dan perangkat lunak tertentu secara berbeda, khususnya sehubungan dengan kinerja eksekusi. Jadi, akan masuk akal untuk mengeksekusi berbagai fase proses simulasi pada mesin komputasi yang berbeda, mungkin dengan pengaturan yang berbeda. Ini menunjukkan sekali lagi bahwa, untuk penggunaan platform yang optimal, pengetahuan yang baik tentang infrastruktur yang mendasarinya adalah penting. Namun,

sebagian besar ilmuwan tidak memiliki - dan tidak tertarik pada - pengetahuan ini. Mereka biasanya lebih memilih untuk fokus pada pengembangan proses simulasi mereka. Oleh karena itu, lingkungan yang memungkinkan para ilmuwan untuk berkonsentrasi pada desain dan aliran eksekusi dari proses simulasi, daripada rincian teknis dari platform TI, sangat penting untuk menghadapi masalah ini.

Fleksibilitas adalah masalah krusial lainnya dalam alur kerja ilmiah. Karena perubahan lingkungan dan perilaku aplikasi yang tidak dapat diprediksi dan dinamis, lingkungan untuk proses simulasi tersebut harus dapat memungkinkan penggunaan sumber daya platform secara transparan. Fleksibilitas yang diperlukan ini memiliki dua konsekuensi utama: di satu sisi, penyesuaian alur kerja itu sendiri mungkin diperlukan, karena hasil perantara yang diterima, mis. menunjukkan bahwa cabang tertentu dari simulasi harus diperhitungkan secara lebih rinci. Di sisi lain, mungkin perlu menyesuaikan infrastruktur layanan, misalnya karena pemantauan eksekusi menunjukkan bahwa beberapa langkah pemrosesan dapat dilakukan dengan cara yang lebih efisien jika diterapkan pada sistem tertentu atau mengungkapkan bahwa beberapa langkah dapat diadaptasi untuk energi efisiensi di lingkungan runtime (misalnya, dengan memutar CPU atau memori untuk bekerja dalam mode hemat energi).

Dari perspektif teknis, masalah ini dapat dihadapi dengan menyediakan sumber daya HPC dalam mode seperti SOA, memungkinkan penggunaan sumber daya tanpa perlu mempertimbangkan detail teknologi yang sesuai. Oleh karena itu, akses yang dapat dioperasikan ke sumber daya melalui teknologi layanan web, yaitu dengan mengacu pada standar WSDL dan SOAP, akan memungkinkan memperoleh akses yang dapat dioperasikan dan abstrak ke sumber daya dengan menyembunyikan, setidaknya sampai tingkat tertentu, kekhasan teknis infrastruktur yang mendasarinya. Berikut ini, sumber daya ini dapat diatur dengan bahasa alur kerja yang canggih, seperti BPEL.

Penggunaan transparan alur kerja dan infrastruktur layanan memerlukan transformasi data otomatis dan penemuan layanan. Tingkat transparansi platform untuk pengguna akhir sangat dituntut dari lingkungan berbasis layanan web umum, dan bagaimanapun belum disediakan oleh semua sistem layanan. Seperti yang telah dibahas sebelumnya, bahasa alur kerja ilmiah konvensional sebagian besar didorong oleh data dan biasanya disediakan dalam bahasa campuran, yang terkadang membutuhkan pengetahuan mendalam tentang detail teknis.

Singkatnya, lingkungan ilmiah dapat mengambil manfaat dari ketangguhan teknologi alur kerja umum yang telah terbukti, khususnya dengan memanfaatkan mekanisme terintegrasi yang telah terbukti dan berkembang untuk pemulihan, penanganan pengecualian, dan kontrol aliran. Namun, lingkungan ilmiah yang paling umum hanya memungkinkan untuk mengubah parameter masukan, tetapi bukan infrastruktur server.

BPEL, sebagai bahasa spesifikasi bisnis berbasis layanan, diterima secara luas dalam industri dan penelitian, dan memiliki banyak manfaat dalam konteks aplikasi HPC kita. Pertama, memungkinkan integrasi aplikasi lama melalui antarmuka layanan web. Selain itu, dukungan alat yang luas disediakan, memungkinkan desain visual alur kerja dan dengan demikian memudahkan pembuatan alur kerja yang kompleks. Khususnya dukungan visual juga

memungkinkan pengguna dengan sedikit latar belakang TI untuk memodelkan proses yang sesuai.

BPEL dapat diterapkan untuk alur kerja ilmiah. Untuk mengisi kesenjangan sehubungan dengan penanganan kumpulan data besar di lingkungan alur kerja ilmiah, perpanjangan BPEL telah diumumkan, yaitu BPEL-D. Ekstensi BPEL-D memungkinkan deskripsi dependensi data dalam proses BPEL. Namun, dependensi ini diterjemahkan ke dalam BPEL standar sebelum dieksekusi, sehingga dependensi ini tidak tersedia selama runtime. Hal ini membuat pendekatan tidak dapat digunakan dalam lingkungan yang dinamis sama sekali. Pendekatan lain untuk menangani sejumlah besar data dalam proses BPEL. Pendekatan ini memungkinkan integrasi referensi ke kumpulan data di seluruh layanan web, dan memungkinkan untuk merujuk hanya ke kumpulan data yang relevan untuk simulasi tertentu, sehingga memungkinkan transfer hanya kumpulan data yang diperlukan ke node komputasi. Namun, pendekatan tersebut tidak menjelaskan bagaimana set data ini harus ditransfer, khususnya untuk protokol transport yang akan digunakan untuk transfer data, juga untuk sumber data yang akan diakses. Biasanya, kumpulan data ini dilindungi: karenanya untuk akses jarak jauh, spesifikasi kumpulan data tentang penggunaan kumpulan data juga harus disediakan. Sebuah arsitektur untuk alur kerja ilmiah. Implementasi referensi saat ini bergantung pada kerangka AXIS2 apache dan tidak memungkinkan untuk abstraksi dari sumber daya terintegrasi dan dengan demikian, tidak memungkinkan untuk evolusi alur kerja ilmiah yang dinamis dan transparan. Oleh karena itu, realisasi bus layanan yang diperluas yang memungkinkan pemanggilan layanan web virtual harus diintegrasikan.

2.3 INFRASTRUKTUR VIRTUALISASI

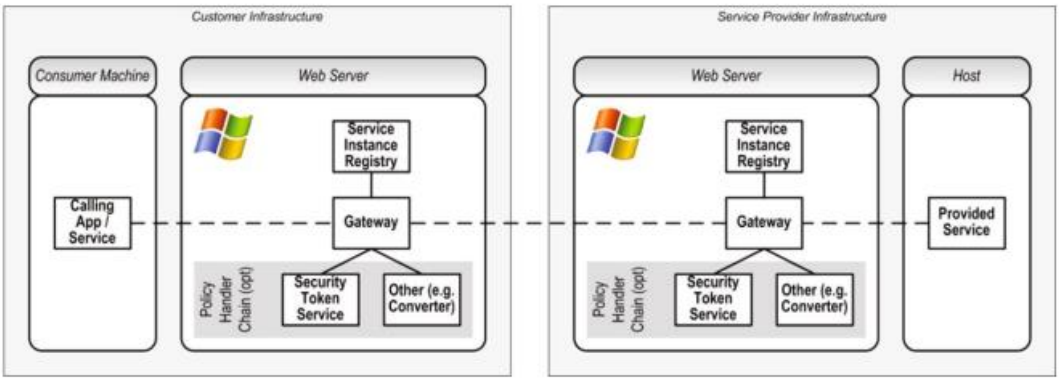
Pada bagian ini arsitektur gateway virtualisasi diperkenalkan. Penjelasan rinci tentang keseluruhan arsitektur gateway virtualisasi, serta realisasi dan evaluasinya. Seperti disebutkan sebelumnya, ada kebutuhan nyata untuk virtualisasi layanan dan akibatnya untuk lapisan abstraksi. Lapisan abstraksi ini beroperasi sebagai layanan perantara antara konsumen dan logika layanan aktual dengan mencegat dan menganalisis pemanggilan dan memetakannya ke layanan yang sesuai, memungkinkan peningkatan yang signifikan dalam fleksibilitas dan dinamika lingkungan terdistribusi yang kompleks. Infrastruktur gateway memungkinkan juga untuk merangkum dan memungkinkan penggantian dinamis sumber daya dalam lingkungan bandara yang kompleks, dengan menerapkan teknologi agen perangkat lunak untuk mewakili sumber daya yang sesuai dan memungkinkan mereka untuk berkomunikasi melalui infrastruktur gerbang virtualisasi. Pemetaan ini juga dapat melibatkan langkah-langkah transformasi yang diperlukan karena gateway virtualisasi tidak berfokus pada deskripsi antarmuka tertentu.

Arsitektur Umum

Karena gateway bertindak sebagai titik masuk tunggal tidak hanya ke logika layanan lokal, tetapi juga berpotensi ke seluruh lingkungan kolaborasi virtual, ia juga menyediakan fungsionalitas untuk mengenkapsulasi layanan.

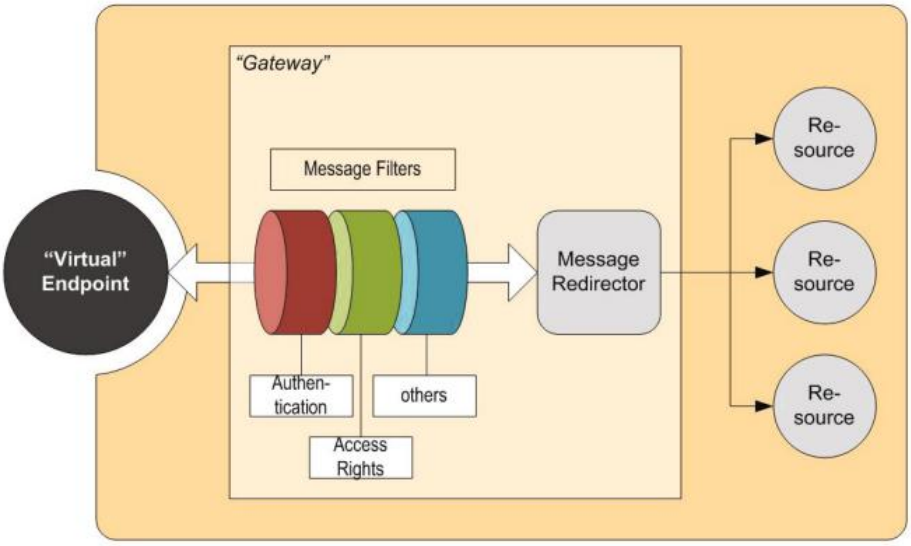
Dengan menggunakan mekanisme "virtualisasi *double-blind*", yaitu, dengan menggunakan gateway di sisi konsumen dan penyedia layanan, dimungkinkan untuk

mengubah sumber daya dan penyedia tanpa memengaruhi aplikasi yang meminta. Gambar 2.1 menyajikan contoh penerapan infrastruktur Gateway. Garis putus-putus menunjukkan link tidak langsung (sebagai gateway memperluas saluran pesan).



Gambar 2.1. Contoh penerapan infrastruktur Gateway

Gambar 2.2 menunjukkan struktur gateway tersebut. Struktur ini memungkinkan penyedia layanan untuk merangkum dan menyembunyikan infrastruktur mereka dengan cara yang juga memungkinkan untuk virtualisasi produk. Dengan gateway yang dapat diperluas, ia memberikan dasar untuk menerapkan kebijakan keamanan, privasi, dan bisnis non-invasif yang terkait dengan transaksi pesan. Dengan pendekatan SOA yang kuat yang dilakukan oleh gateway virtualisasi, struktur tersebut selanjutnya memenuhi persyaratan dampak minimal dan fleksibilitas penerapan maksimum; melalui filternya, ia mendukung dukungan perpesanan standar. Gateway selanjutnya dibangun dengan cara yang memungkinkan partisipasi dalam banyak kolaborasi pada saat yang sama tanpa memerlukan konfigurasi ulang infrastruktur yang mendasarinya.



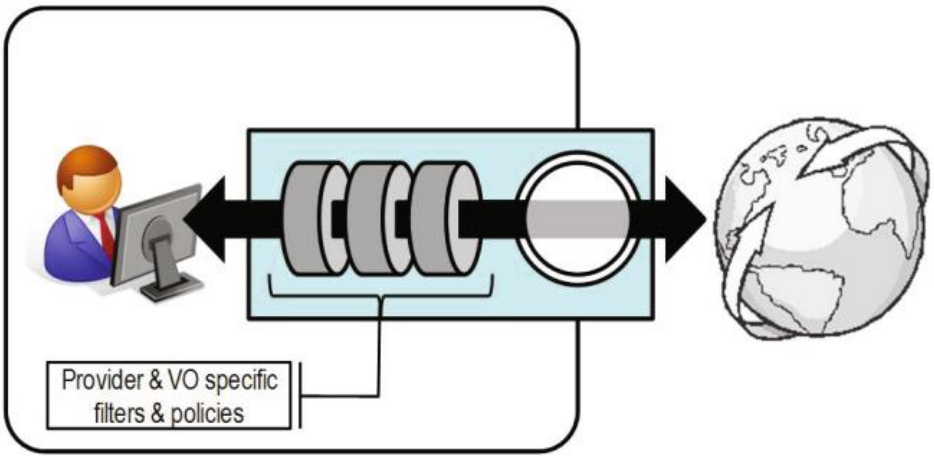
Gambar 2.2. Struktur Gerbang

Akibatnya, teknologi virtualisasi ini bertindak sebagai infrastruktur perpesanan yang ditingkatkan yang dapat melayani tujuan berikut:

- Penegakan kebijakan: Gateway dapat memberlakukan kebijakan pada transaksi pesan karena memungkinkan definisi kriteria yang harus dipenuhi sebelum konsumen potensial diberi wewenang untuk mengakses layanan tertentu. Untuk tujuan ini, poin keputusan kebijakan dapat dimasukkan ke dalam rantai pesan (lihat di bawah). Misalnya, adalah mungkin untuk membedakan konsumen layanan berdasarkan reputasi mereka, sehingga memperkirakan kepercayaan relatif, upaya yang layak diinvestasikan, dll.
- Keamanan pesan, identitas, dan manajemen akses: Idealnya, semua aplikasi klien dan layanan web yang diterapkan mendukung spesifikasi yang sesuai seperti WS-Security, WS-Trust, dan WS-Federation. Idealnya, setiap aplikasi klien harus dapat mengambil token keamanan yang diperlukan untuk akses layanan, dan setiap layanan yang diterapkan harus dapat mengotorisasi permintaan masuk menggunakan model keamanan berbasis klaim dengan otorisasi yang halus. Sayangnya, banyak aplikasi dalam produksi saat ini belum mematuhi prinsip-prinsip ini, dan gateway dapat berfungsi sebagai jalur migrasi menuju adopsi yang lebih luas dari model akses berbasis klaim. Gateway sisi pelanggan dapat mengautentikasi pemohon internal, meminta token keamanan atas nama mereka, dan melindungi (yaitu mengenkripsi) pesan keluar. Gateway sisi layanan dapat bertindak sebagai titik penegakan kebijakan untuk mengautentikasi dan mengotorisasi panggilan masuk. Misalnya gateway dapat membuat sambungan aman ke konsumen layanan sementara aplikasi klien beton tidak mendukung transmisi data yang aman.
- Penerjemahan protokol: Tidak hanya spesifikasi protokol layanan web yang dapat berubah terus-menerus, tetapi juga lingkup protokol yang luas melayani kebutuhan serupa, sehingga menyebabkan masalah interoperabilitas berulang antara pelanggan dan penyedia. Dengan antarmuka statis, konsumen harus memastikan bahwa permintaan layanannya sesuai dengan spesifikasi penyedia.
- Transformasi: Karena gateway menyediakan antarmuka universal untuk layanan dasar, transformasi harus dilakukan sebelum pesan diteruskan ke layanan terkait. Tindakan transformasi dapat dilakukan sebagai langkah lanjutan dari translasi protokol, khususnya mengenai transformasi parameter spesifik layanan.
- Penyaringan dan perlindungan kebocoran informasi: Gateway dapat mendeteksi dan menghapus informasi pribadi dari permintaan, menawarkan pengait untuk memasang mekanisme deteksi dan pencegahan kebocoran informasi.
- *Load balancing & fail over*: Gateway dapat bertindak sebagai penyeimbang muatan. Jika mis. satu layanan sedang banyak digunakan, gateway dapat memutuskan untuk meneruskan permintaan ke layanan ini ke layanan yang setara.
- Perutean: Menggunakan konsep yang mirip dengan DNS, gateway memungkinkan perutean dinamis pesan tergantung pada penyedia yang terdaftar dan tersedia (load balancing). Hal ini memungkinkan penggantian penyedia secara *on-the-fly*, mengingat tidak ada permintaan saat ini yang masih dalam proses (dalam hal ini data yang sesuai akan hilang). Karena gateway memaparkan titik akhir virtual kepada konsumen, dia tidak harus menyesuaikan logika proses, jika penyedia berubah.

- Pemantauan login: Seringkali menarik bagi penyedia layanan untuk melihat versi layanan mana yang masih digunakan oleh pelanggan. Melalui gateway informasi ini juga tersedia.

Gateway penyedia layanan bertindak sebagai titik akhir virtualisasi layanan yang diekspos oleh masing-masing organisasi. Tugas utamanya terdiri dari mencegat pesan masuk dan keluar untuk menegakkan serangkaian kebijakan yang terkait dengan pembatasan hak akses, otentikasi yang aman, transformasi, dll. Dengan demikian memastikan bahwa kebijakan khusus penyedia dan kolaborasi dipertahankan dalam transaksi. Gambar 2.3 menunjukkan gateway dari perspektif pelanggan.



Gambar 2.3. Prinsip Gateway dari Perspektif Pelanggan

Sebagai titik akhir virtual, gateway mampu mengalihkan pesan dari alamat virtual ke lokasi fisik aktual (lokal atau di Internet), sehingga menyederhanakan pengelolaan titik akhir dari sisi klien, yaitu aplikasi/layanan klien tidak terpengaruh oleh perubahan dalam kolaborasi, seperti penggantian penyedia dll. Handler intrinsik dalam saluran gateway karenanya terdiri dari mekanisme resolusi titik akhir yang mengidentifikasi tujuan sebenarnya dari pesan yang diberikan.

Untuk mengumumkan layanan, penyedia layanan menerbitkan definisi antarmuka layanan virtual (WSDL). Panggilan masuk dicegat oleh gateway untuk mengubah pesan-pesan ini sesuai dengan persyaratan implementasi layanan. Untuk tujuan ini gateway mengakses basis pengetahuan yang berisi semua informasi yang diperlukan seperti mis. pemetaan nama virtual ke titik akhir layanan konkret dan transformasi nama metode dan parameter. Karena gateway dapat merutekan ke sejumlah titik akhir fisik, pendekatan ini memungkinkan penyedia untuk mengekspos beberapa dokumen WSDL untuk layanan yang sama, serta sebaliknya, yaitu untuk mengekspos beberapa layanan melalui satu antarmuka layanan web (virtual). Hal ini memungkinkan tidak hanya untuk membedakan antara berbagai contoh (berpotensi khusus pengguna) dari satu layanan web, seperti yang saat ini dieksploitasi oleh kumpulan layanan cloud, tetapi juga memungkinkan penyedia layanan untuk secara dinamis memindahkan contoh layanan antar sumber daya, serta menyusun layanan baru melalui kombinasi aplikasi yang ada, tanpa perlu menulis ulang logika aplikasi. Oleh karena itu juga dimungkinkan untuk menyediakan layanan secara dinamis - misalnya, dengan mengumumkan

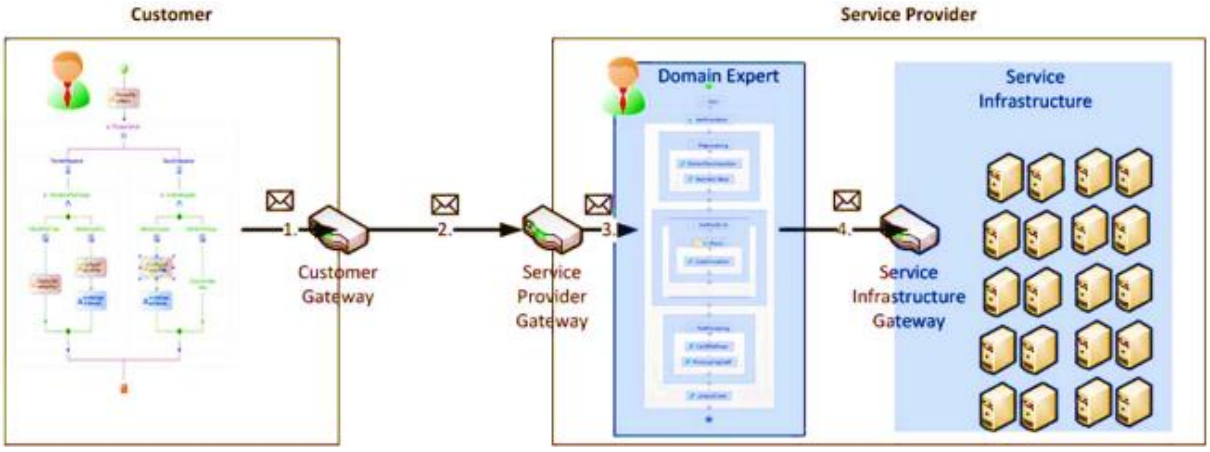
antarmuka virtual baru, atau dengan mengubah logika layanan tanpa perlu mengubah antarmuka.

Aspek-aspek ini menyederhanakan manajemen layanan secara signifikan, karena masalah keharusan melayani instance individual dan perwakilannya dihapus. Misalnya, jika pendekatan digunakan dalam mode seperti cloud, pembaruan yang dimulai di situs pemuliaan akan secara otomatis menyebar ke seluruh jaringan dari waktu ke waktu.

Menerapkan Infrastruktur Gateway ke Domain yang Berbeda

Pada bagian ini kami menjelaskan bagaimana gateway dapat diterapkan untuk menghadapi masalah yang menantang tentang penyediaan produk layanan TI yang kompleks. Secara khusus, kami menggambarkan bagaimana Infrastruktur Gateway dapat diterapkan pada lingkungan konkret mengikuti pendekatan hierarkis. Proses ini memungkinkan untuk konsentrasi keahlian dalam domain terkait, sementara memungkinkan ahli dari domain ini untuk menggunakan dan menyesuaikan layanan dari domain lain dengan mudah. Pendekatan seperti itu digambarkan pada Gambar 2.4. Dalam konteks ini Infrastruktur Gateway telah diterapkan ke domain berikut:

- Pelanggan layanan;
- Penyedia layanan;
- Infrastruktur layanan (bertindak sebagai gateway lokal yang tidak diharapkan untuk penggunaan eksternal).



Gambar 2.4. Menerapkan infrastruktur virtualisasi ke domain yang berbeda

Dalam lingkungan ini, pelanggan layanan telah merancang alur kerja lokal untuk mengatur berbagai layanan yang disediakan secara lokal atau jarak jauh. Seperti yang digambarkan, semua pemanggilan dilakukan melalui titik akhir layanan virtual, yang dicegat oleh gateway pelanggan yang tepat. Pada langkah selanjutnya, logika gateway virtualisasi memutuskan ke tujuan mana dan dalam format apa pesan harus dialihkan. Karena pemanggilan ini dilakukan melalui titik akhir layanan virtual, perubahan titik akhir layanan konkret sama sekali tidak memengaruhi alur kerja pelanggan, karena, selain perutean transparan dari pesan yang sesuai, gateway juga memungkinkan transformasi pesan transparan. Jadi, dalam hal mengubah titik akhir layanan untuk titik akhir layanan virtual, gateway memungkinkan integrasi aturan untuk memetakan pemanggilan umum ke

antarmuka layanan baru. Secara khusus, untuk perancang alur kerja pelanggan, tidak relevan jika layanan dijalankan secara lokal atau jarak jauh. Perancang proses dapat berkonsentrasi pada desain logika alur proses tanpa perlu mempertimbangkan infrastruktur layanan yang konkret.

Domain penyedia layanan dibagi menjadi dua sub-domain, karena penyedia layanan memberikan layanan dan infrastruktur TI yang kompleks. Tentu saja, pendekatan yang disajikan juga memungkinkan integrasi penyedia infrastruktur eksternal. Dalam hal ini, Gerbang Infrastruktur Layanan akan berlokasi di penyedia layanan infrastruktur pihak ketiga. Namun, ini tidak mempengaruhi orkestrasi alur kerja dari sisi ahli domain atau penyedia layanan, karena pemanggilan infrastruktur layanan juga dikemas dalam infrastruktur gateway dengan menggunakan endpoint layanan virtual yang diumumkan.

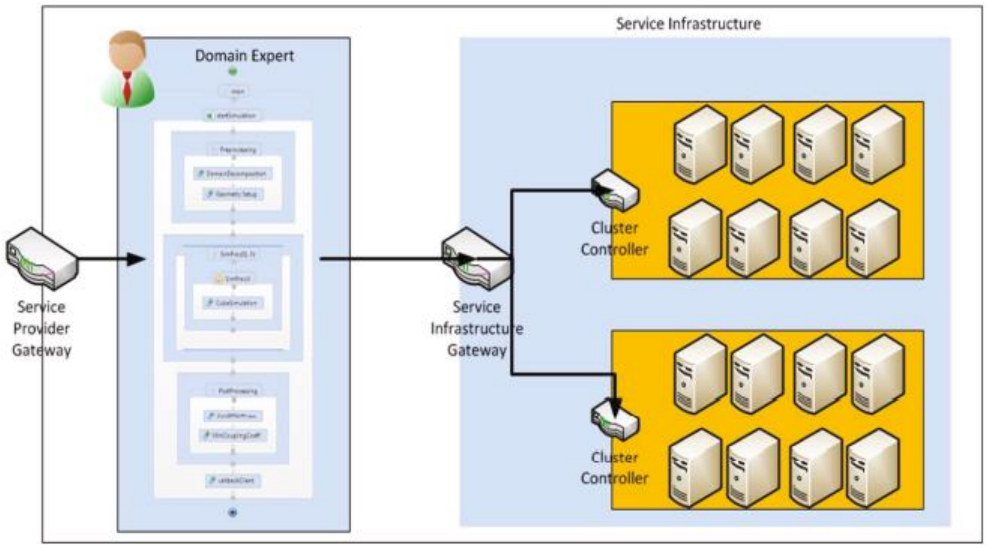
Produk layanan TI yang kompleks disediakan dengan mengizinkan pakar domain (dalam kasus kami, pakar simulasi medis) untuk merancang alur kerja yang mencerminkan langkah-langkah yang diperlukan dengan mengatur layanan yang dibutuhkan. Semua layanan ini terintegrasi dalam alur kerja melalui titik akhir layanan virtual, sehingga mencerminkan fungsionalitas layanan dan bagaimana layanan ini harus dijalankan. Tidak ada detail teknis lainnya yang harus dipertimbangkan pada tahap ini untuk desainer alur kerja.

Terakhir, Gateway Infrastruktur Layanan memungkinkan pemetaan permintaan ke infrastruktur layanan. Berikut ini, kami menyajikan pemrosesan infrastruktur gateway hierarkis ini dengan mengilustrasikan pemanggilan layanan konkret yang dilakukan dari pelanggan layanan.

a. Sebagai bagian dari alur kerja pelanggan yang mencerminkan simulasi wilayah tertentu dari tubuh manusia, proses simulasi tulang cancellous harus dipicu. Mesin alur kerja pelanggan memanggil titik akhir layanan virtual yang sesuai.
b. Pesan yang sesuai dicegat oleh gateway pelanggan yang meneruskan permintaan ini ke penyedia layanan yang sesuai, yang memiliki kontrak dengan perusahaan pelanggan untuk jenis simulasi ini. Perancang alur kerja serta lingkungan eksekusi tidak mengetahui penyedia layanan mana yang akan memproses langkah dalam alur kerja ini. Biasanya, pelanggan layanan tidak ingin direpotkan dengan detail ini, melainkan ingin menerima hasil percobaan.
c. Pakar domain dari penyedia layanan merancang proses simulasi tulang cancellous dengan menerapkan fungsi, aplikasi, dan aliran data yang diperlukan (dalam urutan yang benar). Dengan memanfaatkan Gateway Infrastruktur Layanan internal, pakar domain tidak harus memperhitungkan detail teknis infrastruktur layanan. Pakar domain diaktifkan untuk hanya mengatur titik akhir layanan virtual dalam proses simulasi, yang diterapkan secara internal. Setelah menerima permintaan simulasi dari pelanggan, diperiksa apakah pelanggan diizinkan untuk memproses permintaan ini. Kemudian, pemanggilan diteruskan ke mesin alur kerja yang menghosting layanan yang diminta.
d. Terakhir, aktivitas alur kerja lokal penyedia layanan ditransmisikan melalui Infrastruktur Layanan ke node komputasi yang sesuai.

Gambar 2.5 menggambarkan secara rinci bagaimana pemetaan permintaan ke infrastruktur layanan ditangani. Gateway virtualisasi telah direalisasikan menggunakan .NET 3.5, dengan mengimplementasikan seluruh perutean dan mengelola logika. Namun, sebagian besar lingkungan infrastruktur TI tidak memungkinkan penyebaran Layanan .NET. Jadi, untuk mengarahkan pemanggilan aplikasi dan layanan pada Infrastruktur Layanan TI, telah direalisasikan Cluster Controller tambahan. Cluster Controller diimplementasikan sebagai aplikasi gSOAP berbobot ringan, memungkinkan eksekusi operasi sistem lokal yang dapat dipicu melalui antarmuka layanan web. Selain itu, Pengontrol Cluster memungkinkan untuk meneruskan referensi ke set data ke node komputasi dan cluster, masing-masing. Referensi ini meliputi:

- Protokol transfer data dan lokasi kumpulan data yang diperlukan.
- Secara opsional, kueri yang memungkinkan pemilihan elemen spesifik dari kumpulan data. Ini mengurangi jumlah data yang akan ditransfer ke node komputasi.
- Informasi akun diperlukan untuk mengakses kumpulan data.



Gambar 2.5. Menerapkan gateway untuk menangani infrastruktur layanan lokal

Referensi-referensi ini memungkinkan manajemen yang cerdas dari kumpulan data besar karena gateway menerapkan perutean yang dikelola sendiri dari informasi dan pesan yang diperlukan mengikuti pendekatan berbasis SOA.

2.4 PENJADWALAN DAN PENERAPAN PEKERJAAN SADAR ENERGI

Seperti dijelaskan dalam Bagian 2.2, teknologi alur kerja umum yang dibuat di berbagai domain dapat diterapkan untuk menyediakan dan mengintegrasikan sumber daya HPC dalam hal SOA. Secara khusus, BPEL memfasilitasi orkestrasi, pemulihan, penanganan pengecualian, dan kontrol alur kerja ilmiah. Namun, dengan meningkatnya kompleksitas dalam domain HPC, alur kerja menggunakan sumber daya TI yang terpusat/terdistribusi heterogen, seperti CPU, memori, jaringan, dan penyimpanan data secara besar-besaran. Akibatnya, alur kerja yang kompleks menghabiskan banyak energi. Alur kerja hijau dibayangkan untuk meminimalkan beban lingkungan sambil memaksimalkan efisiensi energi alur kerja tanpa mengorbankan

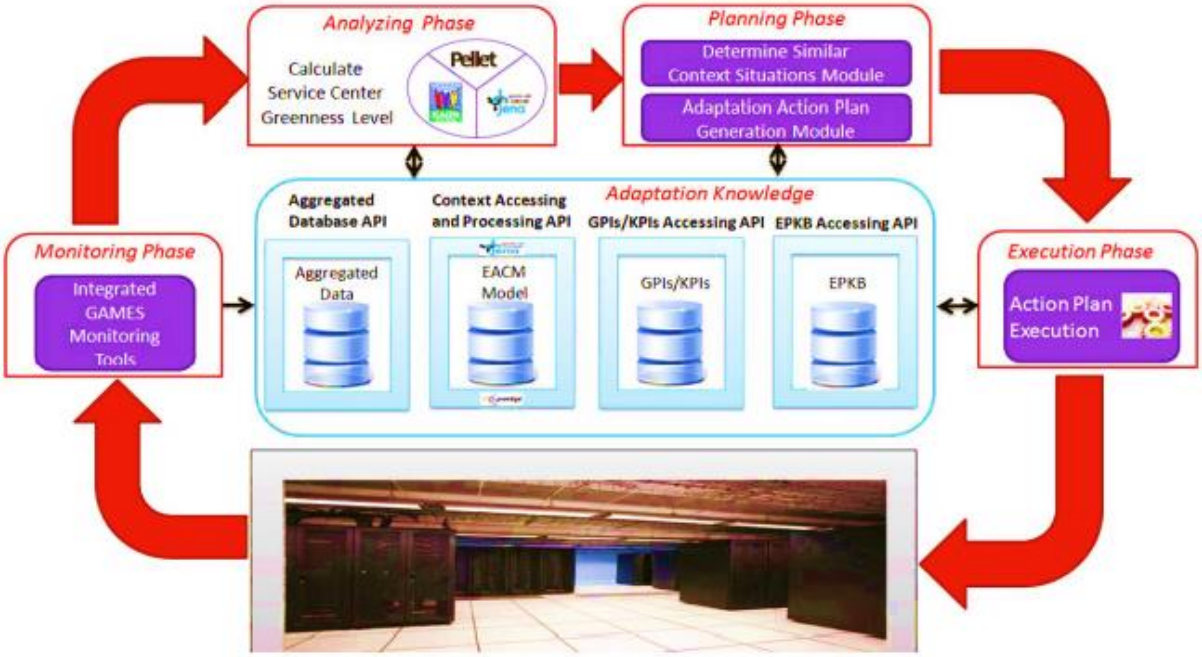
kendala kinerja dan persyaratan QoS tertentu. Karena konsumsi energi ditentukan oleh perangkat keras yang mendasari yang dialokasikan oleh alur kerja, heuristik yang efektif dan mekanisme untuk adaptasi dinamis dari penjadwalan pekerjaan dan penerapan sesuai dengan pemanfaatan sumber daya saat ini diperlukan untuk mencapai efisiensi energi yang lebih baik. Misalnya beban kerja tugas dalam alur kerja dapat dikonsolidasikan dan kemudian sumber daya yang tidak terpakai dapat dialihkan ke kondisi daya rendah, tingkat kinerja rendah, atau bahkan dimatikan untuk menghemat energi. Berikut ini kami akan memperkenalkan Global Control Loop yang diusulkan dan dikembangkan dalam proyek GAMES dan sepenuhnya kompeten untuk alokasi sumber daya sadar energi yang disebutkan di atas untuk alur kerja.

Di pusat layanan virtual, biasanya sumber daya komputasi disediakan secara berlebihan untuk dapat menangani beban puncak. Global Control Loop mengeksploitasi fitur ini dengan merasakan perubahan beban kerja, secara proaktif mendeteksi pusat layanan atas sumber daya komputasi TI yang disediakan dan mengidentifikasi/melaksanakan tindakan yang tepat untuk menyesuaikan penggunaan sumber daya komputasi TI dengan beban kerja yang masuk. Putaran Kontrol Global bertujuan untuk: (i) menganalisis data energi/kinerja pusat layanan untuk mendeteksi situasi tersebut di mana waktu desain yang ditetapkan Tingkat Hijau dan Indikator Kinerja Utama tidak tercapai, (ii) memutuskan tindakan adaptasi untuk menegakkan GPI/KPI dan (iii) bertindak untuk melaksanakan tindakan adaptasi tersebut.

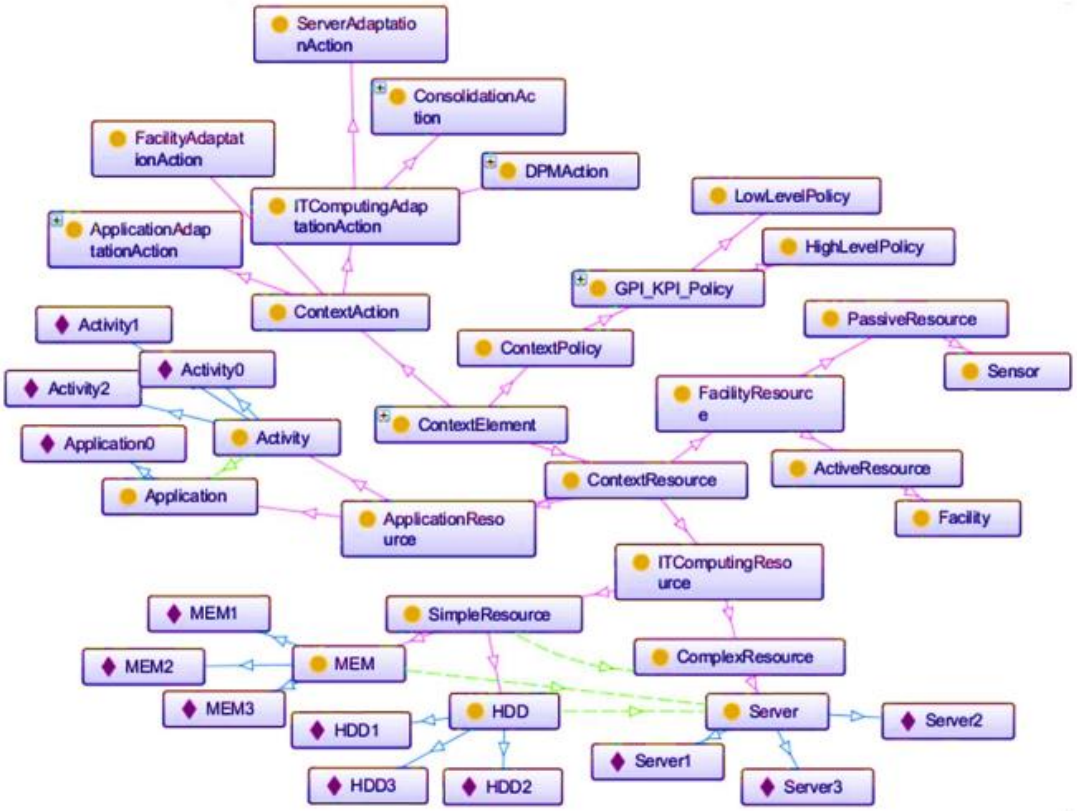
Loop Kontrol Global terletak di atas Infrastruktur virtualisasi sumber daya Penyedia Layanan. Oleh karena itu, loop Kontrol Global berinteraksi dengan Infrastruktur Pemantauan GAMES dan dengan Layanan Registri Instance infrastruktur Gateway Virtualisasi untuk mengarahkan penerapan dan parameterisasi pekerjaan HPC yang sesuai. Prosedur ini memungkinkan Global Control Loop menerapkan dua jenis tindakan adaptasi: (i) tindakan konsolidasi beban kerja seperti penerapan dan migrasi pekerjaan dan (ii) tindakan manajemen daya dinamis di tingkat server seperti membangunkan atau mematikan server. Misalnya, jika Pelanggan mengeluarkan permintaan untuk layanan virtual ke Infrastruktur Penyedia, Global Control Loop sisi Penyedia akan memeriksa layanan yang paling sesuai untuk permintaan terkait, dengan mempertimbangkan efisiensi energi dari pelaksanaan seluruh alur kerja, dan mengonfigurasi Registri Instans Layanan yang memungkinkan eksekusi transparan dan penyiapan layanan terbaik yang sesuai secara transparan.

Proses keputusan adaptasi kesadaran energi Global Control Loop diimplementasikan menggunakan MAPE (Monitoring, Analysis, Planning/Deciding and Execution) berbasis loop umpan balik kontrol penyesuaian diri (lihat Gambar 2.6).

Fase Pemantauan bertanggung jawab untuk mengumpulkan dan merepresentasikan secara terprogram data terkait energi/kinerja. Data terkait energi/kinerja pusat layanan dikumpulkan menggunakan Alat Pemantauan GAMES Terintegrasi dan disimpan dalam Basis Data Agregat. Data dari Basis Data Agregat bersifat heterogen dan sulit untuk diproses secara otomatis. Untuk mengatasi masalah ini, model EACM (Energy Aware Context Model) telah dikembangkan untuk merepresentasikan data terkait energi/kinerja dengan cara yang dapat dieksekusi melalui ontologi (lihat Gambar 2.7).



Gambar 2.6. Fase MAPE adaptasi mandiri Global Control Loop



Gambar 2.7. Memodelkan data konteks terkait energi/kinerja pusat layanan menggunakan ontologi model EACM. Kotak bertanda lingkaran mewakili konsep model EACM sedangkan kotak bertanda berlian mewakili contoh konsep

Dalam model EACM, data konteks terkait energi/kinerja direpresentasikan menggunakan tiga konsep utama: Sumber Daya Konteks, Tindakan Konteks, dan Kebijakan Konteks. Konsep Sumber Daya Konteks mendefinisikan entitas fisik atau virtual yang menghasilkan dan/atau memproses data konteks terkait energi. Tindakan Konteks menentukan serangkaian tindakan adaptasi yang dapat dijalankan oleh Global Control Loop pada waktu berjalan untuk menegakkan sasaran efisiensi energi pusat layanan. Kebijakan Konteks menentukan sasaran efisiensi energi pusat layanan melalui kumpulan waktu desain GPI/KPI yang ditentukan.

Untuk mengukur tingkat kehijauan pusat layanan, kami telah mendefinisikan konsep entropi situasi konteks pusat layanan (E). Entropi adalah metrik yang menetapkan tingkat situasi konteks pusat layanan saat ini untuk mematuhi semua GPI/KPI yang ditentukan. Nilai entropi situasi pusat layanan (S) dihitung menggunakan relasi berikut:

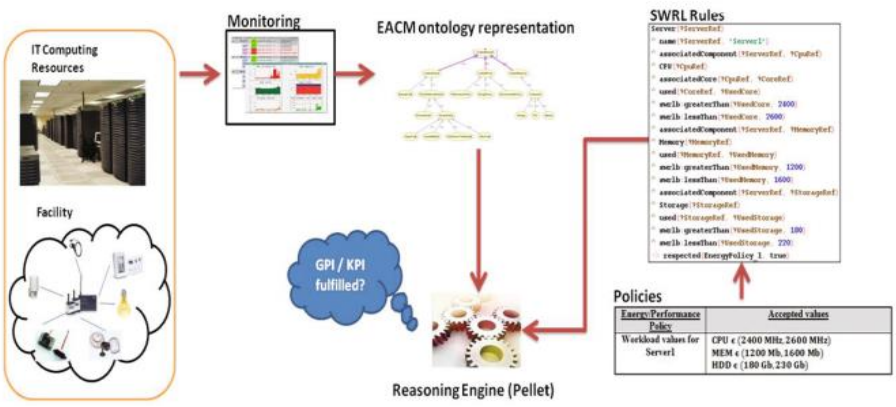
Persamaan 2.1

$$Es = \sum_{i} w_{(Pi)} \sum_{j} w_{(r_{ij})} \cdot v_{(r_{ij})}$$

di mana: (i) $w_{(Pi)}$adalah bobot kebijakan GPI/KPI i, (ii) $w_{(r_{ij})}$.adalah bobot sumber daya konteks pusat layanan j dalam kebijakan GPI/KPI i dan (iii) $v_{(r_{ij})}$ adalah penyimpangan antara nilai yang dicatat oleh sumber konteks j dan nilai yang diterima yang ditentukan oleh kebijakan (i).

Nilai entropi digunakan untuk memicu fase perencanaan Global Control Loop sebagai berikut: jika nilai entropi di atas ambang batas yang telah ditentukan, tingkat kehijauan pusat layanan dapat diterima dan tindakan adaptasi tidak diperlukan, jika tidak, proses perencanaan adaptasi akan dimulai.

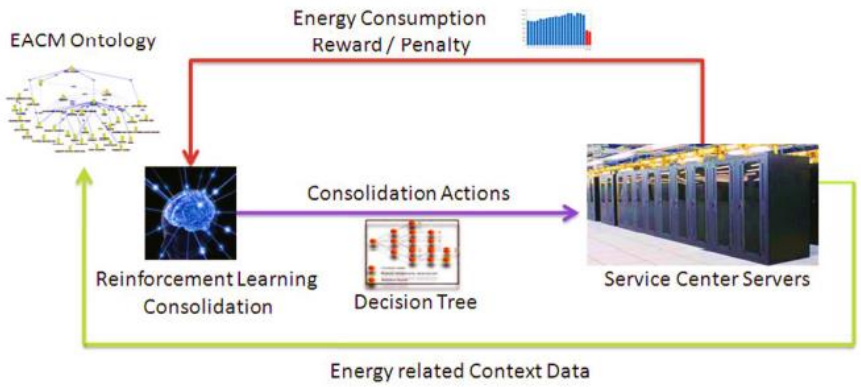
Fase Analisis bertanggung jawab untuk menginterpretasikan data energi/kinerja yang disediakan oleh fase pemantauan, yang bertujuan untuk mendeteksi situasi di mana level GPI/KPI tidak tercapai dan proses adaptasi perlu dijalankan. Untuk mengevaluasi kebijakan GPI/KPI, mereka direpresentasikan dalam XML dan secara otomatis dikonversi dalam aturan SWRL menggunakan model representasi/evaluasi kebijakan. Representasi SWRL dari aturan GPI/KPI kemudian disuntikkan ke implementasi ontologi instance model EACM dan dievaluasi menggunakan mesin penalaran (lihat Gambar 2.8).



Gambar 2.8. Evaluasi kebijakan GPI/KPI

Tahap Perencanaan bertanggung jawab untuk memutuskan rencana aksi adaptasi yang harus dijalankan. Pada fase ini EPKB (Basis Pengetahuan Praktik Energi) dicari untuk mengidentifikasi situasi yang setara di mana nilai serupa untuk GPI/KPI ditentukan. Jika situasi yang setara ditemukan, rencana aksi adaptasi yang sama diambil.

Jika tidak, urutan tindakan ditentukan melalui algoritma pembelajaran penguatan. Algoritme dimulai dari situasi konteks pusat layanan saat ini dan membangun pohon keputusan dengan mensimulasikan pelaksanaan semua tindakan adaptasi yang tersedia menggunakan hadiah/penalti (lihat Gambar 2.9). Dalam pendekatan kami untuk mensimulasikan pelaksanaan tindakan menyiratkan menciptakan situasi konteks pusat layanan (diwakili oleh contoh EACM) yang sesuai dengan keadaan pusat layanan yang akan dihasilkan sebagai hasil dari penegakan tindakan. Node pohon menyimpan contoh EACM yang menjelaskan situasi konteks terkait energi pusat layanan dan nilai entropi terkaitnya. Jalur pohon antara dua node Node0 dan Noden mendefinisikan urutan tindakan yang dieksekusi mulai dari situasi konteks tersimpan Node0 menghasilkan situasi konteks pusat layanan baru yang disimpan oleh node Noden. Untuk loop kontrol global, jalur entropi minimum dalam pohon keputusan pembelajaran penguatan mewakili urutan tindakan adaptasi terbaik yang akan dieksekusi. Fase Eksekusi atau Bertindak bertanggung jawab untuk menerapkan tindakan yang ditentukan oleh proses perencanaan.



Gambar 2.9. Proses perencanaan tindakan adaptasi berbasis penguatan pembelajaran

Seperti yang ditunjukkan di atas, *Global Control Loop* dirancang dengan tujuan untuk mengidentifikasi dan menjalankan strategi manajemen daya dan kinerja yang paling tepat untuk mengadaptasi sumber daya yang disediakan ke beban kerja yang masuk. Untuk memanfaatkannya dan membuat penyediaan layanan HPC hemat energi, *Global Control Loop* akan diintegrasikan dengan infrastruktur virtualisasi dengan menggabungkannya dengan Gateway Infrastruktur Layanan yang terletak di domain infrastruktur layanan. Interaksi di antara mereka dijelaskan di bawah ini:

- Basis pengetahuan yang digunakan oleh *Global Control Loop* diperluas sebagai registri contoh layanan untuk menyimpan permintaan sumber daya, persyaratan QoS, dan ambang aktivitas GPI/KPI dari alur kerja lokal penyedia layanan serta deskripsi infrastruktur yang mendasarinya

- Di awal alur kerja simulasi diperlukan langkah-langkah tambahan, di mana permintaan penerapan pekerjaan yang sesuai dan respons akan dipertukarkan antara mesin alur kerja lokal dan Global Control Loop
- Pekerjaan simulasi akan diterapkan dan infrastruktur akan diadaptasi secara dinamis oleh Global Control Loop dengan cara yang hemat energi.
- Pemanggilan infrastruktur layanan yang sesuai tetap tidak tersentuh dengan menggunakan titik akhir layanan virtual yang akan dialihkan ke sumber daya sebenarnya oleh Service Infrastructure Gateway.

Untuk meningkatkan konsep dan mengevaluasi keefektifan Global Control Loop, kami melakukan proses BPEL yang terdiri dari serangkaian eksekusi dengan interaksi dengan Global Control Loop dan pemanggilan layanan yang berbeda dengan jejak konsumsi sumber daya yang berbeda, untuk menyelidiki berbagai faktor dampak pada konsumsi energi. Detailnya dijelaskan di bagian selanjutnya.

2.5 CONTOH: ALUR KERJA HPC

BPEL juga dapat digunakan untuk menggambarkan alur kerja ilmiah; namun, realisasi alur kerja semacam ini dengan BPEL masih menghadapi beberapa kesenjangan besar. Oleh karena itu, kami menjelaskan di Bagian 2.3 bagaimana infrastruktur virtualisasi kami dapat mengaktifkan penggunaan BPEL untuk alur kerja tersebut guna mengisi celah ini. Di bagian ini kami menyediakan deskripsi alur kerja skenario penggunaan HPC kami.

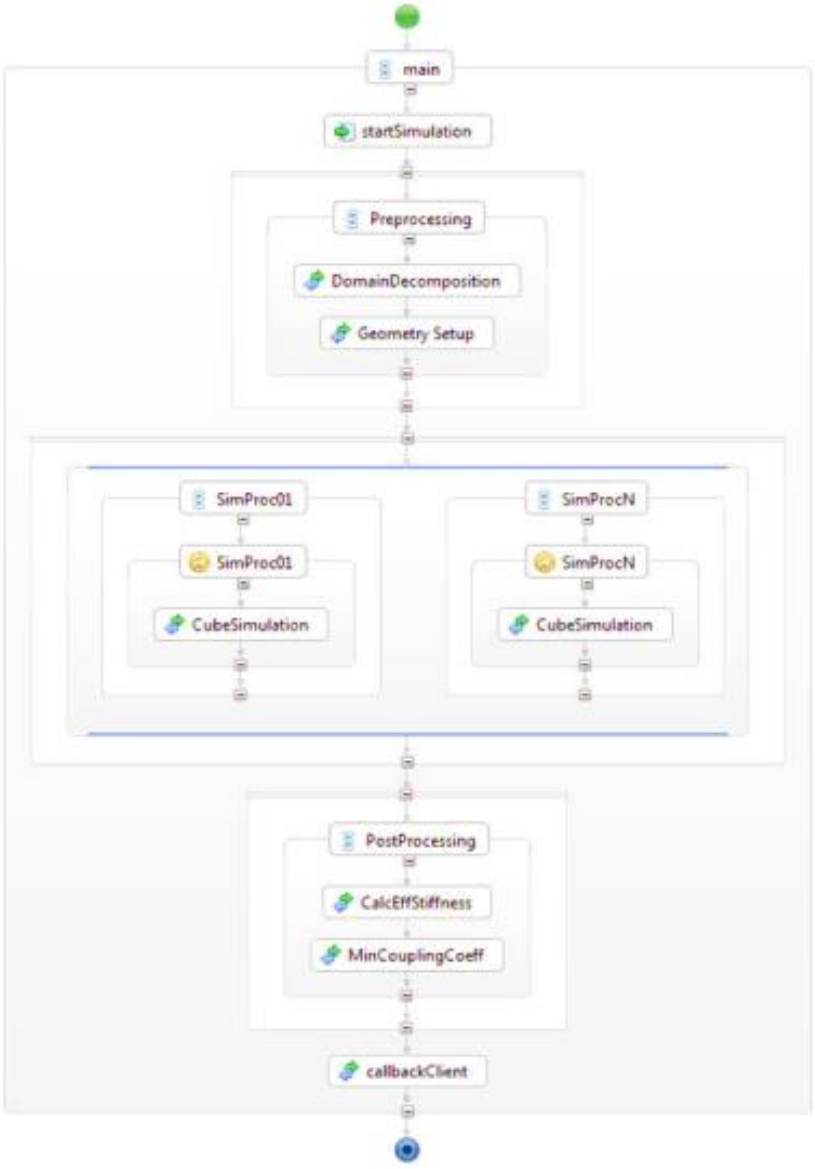
Proses BPEL berikut telah dikembangkan menggunakan desainer Eclipse BPEL terintegrasi. Ini adalah contoh bagaimana editor grafis standar dapat digunakan untuk mengatur proses yang kompleks dan membuatnya tersedia sebagai layanan yang mudah digunakan untuk pelanggan eksternal. Dengan melanjutkan cara ini, para ilmuwan dapat berkonsentrasi pada simulasi tanpa perlu memperhitungkan infrastruktur yang mendasarinya. Gambar 2.10 adalah representasi grafis dari proses BPEL ini dari sudut pandang desainer Eclipse BPEL dalam simulasi tulang cancellous.

Proses dimulai dengan menerima pesan pemanggilan melalui antarmuka layanan web dengan memicu metode startSimualtion. Selanjutnya, kegiatan proses simulasi yang telah ditentukan dijalankan, yaitu:

- Pemrosesan Awal;
- Simulasi;
- Pengolahan pasca.

Dari hasil simulasi.

Fungsionalitas yang diperlukan dari setiap langkah disediakan oleh aplikasi terpisah, yang dapat dipanggil melalui antarmuka layanan virtual. Pemanggilan ini dipetakan pada langkah selanjutnya ke titik akhir layanan konkret, atau mungkin diteruskan ke pihak eksternal. Namun, ilmuwan yang merancang proses ini tidak harus mengetahui lokasi layanan ini. Dia hanya bisa mengatur fungsi yang dibutuhkan dalam proses BPEL; infrastruktur virtualisasi menjaga agar doa dikirim ke layanan yang benar.



Gambar 2.10. Proses BPEL untuk simulasi tulang cancellous

2.6 EVALUASI

Untuk menentukan potensi pengoptimalan *Global Control Loop*, proses BPEL yang meminta layanan berbeda dengan jejak konsumsi sumber daya yang berbeda telah diterapkan pada testbed GAMES, dengan menggunakan lingkungan OpenNebula *Cloud Computing*. Oleh karena itu, Gambar 2.11 menggambarkan perbandingan konsumsi energi dari dua mode eksekusi ini.

Kami menghitung dan menunjukkan di sini juga kesalahan standar dari konsumsi energi dari beberapa uji coba, yang merupakan standar deviasi dari rata-rata sampel dan dengan demikian memberikan ukuran penyebarannya. Sekitar 68% dari semua rata-rata sampel akan berada dalam satu kesalahan standar dari rata-rata populasi (dan 95% dalam dua kesalahan standar). Kesalahan standar menunjukkan keakuratan rata-rata sampel dibandingkan dengan rata-rata populasi. Semakin kecil kesalahan standar, semakin kecil penyebarannya dan semakin besar kemungkinan rata-rata sampel mendekati rata-rata

populasi. Seperti yang ditunjukkan pada Gambar 2.11, ada dua puncak positif pada sekitar 0:27 dan 1:33, yang disebabkan oleh penerapan dan booting 5 VM. Dua puncak negatif sekitar pukul 1:00 dan 1:57 disebabkan oleh tidak diterapkannya VM. Dalam eksekusi yang diaktifkan GAMES, konsumsi daya jelas lebih rendah daripada eksekusi normal, karena node komputasi yang tidak diperlukan dimatikan dan dibangunkan kembali saat tugas yang baru diterapkan datang melalui *Global Control Loop*. Perlu dicatat bahwa dalam pengujian kami, hanya 5 VM yang diterapkan di testbed, dan maksimum 5 node dari 18 node digunakan untuk menghosting VM dan menjalankan layanan. Jelas, peningkatan dapat dibuat lebih signifikan jika node testbed telah digunakan sepenuhnya, tetapi ini akan dilakukan di pengujian mendatang.

Gambar 2.11. Perbandingan konsumsi daya rata-rata

Ringkasnya, *Global Control Loop* mengatasi masalah konsumsi energi tinggi pusat layanan dengan mengonsolidasikan beban kerja pusat layanan sehingga sumber daya komputasinya digunakan secara lebih efisien dan akibatnya konsumsi energi total berkurang. Implementasi *Global Control Loop* untuk evaluasi pertama ini tidak memeriksa dan mematikan server yang tidak menjalankan beban kerja sebelum tindakan konsolidasi diambil. Bahkan dalam keadaan seperti ini, hasil uji coba pertama menunjukkan potensi penghematan energi dari *Global Control Loop* cukup menjanjikan.

2.7 PENUTUP

Dalam bab ini, kami telah menyajikan pendekatan baru yang memungkinkan penyediaan layanan HPC yang kompleks dengan cara yang mudah digunakan dan transparan, sembari mempertimbangkan efisiensi energi yang sesuai dari pekerjaan beton yang dijalankan. Kami telah menjelaskan bagaimana infrastruktur virtualisasi layanan kami dapat digabungkan dengan kerangka GAMES, memungkinkan mekanisme kemudi transparan dengan mempertimbangkan konsumsi energi saat ini dari layanan terkait.

Seperti yang ditunjukkan oleh evaluasi pertama kerangka GAMES dalam lingkungan cloud klasik, *Global Control Loop* yang disajikan dapat memiliki dampak yang signifikan terhadap konsumsi energi seluruh pusat. Pada langkah selanjutnya, *Global Control Loop* ini akan disempurnakan dan digabungkan dengan Infrastruktur Gateway Virtualisasi.

Arah penelitian lainnya adalah studi tentang bagaimana struktur proses alur kerja dapat divariasikan (evolusi proses), mengingat bagaimana perubahan dapat membuat eksekusi proses lebih hemat energi, tetap mempertahankan fungsionalitas asli dan QoS yang diperlukan untuk proses tersebut.

Terakhir, karena pendekatan sekarang hanya mempertimbangkan satu proses bisnis per waktu yang berjalan di pusat data, pekerjaan di masa mendatang akan menyelidiki bagaimana situasi yang lebih kompleks dapat dikelola, di mana beberapa proses simulasi atau aktivitas HPC yang lebih kompleks berjalan secara bersamaan.

BAB 3
PERMODELAN MAKRO KONSUMSI DAYA UNTUK SERVER

Untuk mewujudkan masyarakat ramah lingkungan, total konsumsi daya listrik sistem informasi harus dikurangi. Dalam bab ini, kita membahas model baru konsumsi daya server dalam sistem terdistribusi. Kami pertama-tama mengklasifikasikan aplikasi terdistribusi dari sistem informasi menjadi tiga jenis: perhitungan (CP), komunikasi (CM), dan penyimpanan (ST) jenis aplikasi. Kami pertama-tama mengukur pada tingkat makro konsumsi daya seluruh server tempat jenis aplikasi dilakukan. Kami tidak mempertimbangkan berapa banyak daya listrik yang dikonsumsi setiap komponen perangkat keras seperti CPU pada tingkat mikro. Konsumsi daya server tidak hanya bergantung pada komponen perangkat keras tetapi juga komponen perangkat lunak. Berdasarkan eksperimen, kami mengimplikasikan model konsumsi daya server untuk menjalankan jenis proses aplikasi CP, CM, dan ST pada level makro. Kemudian, kami mengusulkan algoritme untuk memilih server dalam satu set server sehingga tidak hanya persyaratan QoS *(Quality of Service)* seperti waktu respons dan batasan tenggat waktu yang berlaku tetapi juga konsumsi daya total server dikurangi berdasarkan konsumsi daya model. Kami mengevaluasi algoritme pemilihan dibandingkan dengan algoritme tradisional seperti *algoritme round robin* dalam hal konsumsi daya total dan kepuasan persyaratan QoS.

3.1 PENDAHULUAN

Dalam sistem informasi terukur seperti peer-to-peer (P2P) dan sistem *cloud computing,* sejumlah besar komputer server saling terhubung dan mengkonsumsi daya listrik. Untuk mewujudkan aplikasi terdistribusi, berbagai jenis algoritma terdistribusi sejauh ini dikembangkan. Misalnya, algoritma untuk mengalokasikan sumber daya komputasi seperti CPU dan memori ke proses dan untuk menyinkronkan beberapa proses yang saling bertentangan dibahas dalam hal biaya dan kinerja. Di sisi lain, lebih penting mewujudkan sistem informasi sadar energi untuk mengurangi total konsumsi daya listrik.

Dalam satu pendekatan untuk mengurangi konsumsi daya sistem informasi, konsumsi daya setiap perangkat keras seperti CPU dan memori dicoba untuk dikurangi. Misalnya, CPU hemat energi diproduksi oleh Intel dan AMD. Bahkan jika konsumsi daya setiap perangkat keras dapat dibuat jelas, sulit untuk memperkirakan total konsumsi daya seluruh komputer karena konsumsi daya bergantung pada jenis komponen perangkat lunak seperti sistem operasi dan proses aplikasi yang dilakukan. Dalam bab ini, penulis lebih suka mempertimbangkan berapa banyak daya elektronik yang dikonsumsi seluruh server pada tingkat makro. Kami ingin membahas pendekatan tingkat makro baru untuk mengurangi konsumsi daya server. Penulis pertama-tama mengukur berapa banyak daya listrik yang dikonsumsi seluruh server untuk menjalankan aplikasi dengan menggunakan meteran daya UWMeter di mana konsumsi daya server dapat diukur setiap seratus milidetik [J/0,1 detik]. Ada berbagai aplikasi terdistribusi seperti aplikasi Web, aplikasi database, dan sistem file Google. Kami mengklasifikasikan aplikasi menjadi tiga jenis: perhitungan (CP), komunikasi

(CM), dan penyimpanan (ST) jenis aplikasi. Dalam aplikasi CP seperti komputasi ilmiah, server terutama menggunakan sumber daya CPU untuk memproses permintaan yang dikeluarkan oleh klien. Dalam aplikasi CM, server terutama mengkonsumsi sumber daya jaringan. Di sini, sejumlah besar data ditransmisikan ke klien seperti aplikasi file transfer protocol (FTP). Dalam aplikasi ST, sumber daya penyimpanan seperti hard disk drive paling banyak digunakan. Di sini, proses membaca dan menulis file di drive penyimpanan seperti hard disk drive.

Pada bab ini, kita membahas bagaimana mengurangi konsumsi daya total server untuk melakukan jenis proses aplikasi dalam sistem client-server. Di sini, klien pertama-tama mengeluarkan permintaan ke penyeimbang muatan. Penyeimbang beban memilih server di kumpulan server dan kemudian meneruskan permintaan ke server. Setelah menerima permintaan, server melakukan permintaan sebagai proses aplikasi dan mengirimkan respons ke klien. Server terutama mengkonsumsi daya untuk menangani permintaan seperti permintaan halaman Web dibandingkan dengan klien. Sangat penting untuk mengurangi konsumsi daya server. Pertama, kita membahas sepasang model komputasi sederhana dan model konsumsi daya sederhana dari server untuk aplikasi CP, yang berasal dari hasil percobaan. Model konsumsi daya sederhana adalah model dua keadaan.

Di sini, daya listrik server dikonsumsi secara maksimal jika setidaknya satu proses aplikasi dilakukan. Jika tidak ada proses yang dilakukan, yaitu server dalam keadaan diam, server mengkonsumsi daya minimum. Selain itu, tingkat perhitungan server maksimum, yaitu dengan siklus jam maksimum jika setidaknya satu proses dilakukan. Ini adalah model komputasi sederhana. Berdasarkan model komputasi sederhana, kami membahas bagaimana memperkirakan waktu ketika setiap proses aplikasi saat ini berakhir di server. Kami membahas algoritma berbasis kelonggaran konsumsi daya (PCLB) untuk memilih server untuk setiap permintaan dari klien di kumpulan server yang mungkin. Dalam algoritma PCLB, server yang mengkonsumsi konsumsi daya minimum dipilih untuk setiap permintaan dari klien. Kami mengevaluasi algoritma PCLB dalam hal konsumsi daya total dan waktu tempuh dibandingkan dengan algoritma dasar round-robin (RR).

Selanjutnya, kita membahas model konsumsi daya server untuk menjalankan aplikasi CM di mana server mengirimkan file ke klien. Di sini, ada dua jenis model transmisi file: model terikat server dan model terikat klien. Dalam model yang terikat server, jumlah total tingkat penerimaan maksimum dari semua klien saat ini dari server lebih besar daripada tingkat transmisi efektif server. Oleh karena itu, semakin banyak klien mengirimkan permintaan ke server, semakin lama waktu yang dibutuhkan untuk mengirimkan file ke klien. Di sisi lain, jumlah total tingkat penerimaan maksimum semua klien lebih kecil daripada tingkat transmisi efektif server dalam model terikat klien. Oleh karena itu, setiap klien dapat menerima file dengan tingkat penerimaan maksimum. Dari hasil percobaan, kami memperoleh properti signifikan dari model konsumsi daya bahwa tingkat konsumsi daya server untuk mengirimkan file meningkat secara linier untuk total tingkat transmisi. Berdasarkan model transmisi file dan model konsumsi daya, kami mengusulkan algoritme berbasis konsumsi daya yang diperluas (EPCB) untuk memilih server di kumpulan server FTP yang memungkinkan. Di sini, server dengan konsumsi daya minimum untuk mengirimkan file ke setiap klien dipilih untuk setiap

permintaan dari klien. Kemudian, kami mengevaluasi algoritme EPCB dalam hal konsumsi daya total dan waktu tempuh dibandingkan dengan algoritme RR.

Terakhir, kami membahas model konsumsi daya server untuk melakukan proses aplikasi di aplikasi ST. Kami mengukur tingkat konsumsi daya server tempat file di drive penyimpanan dimanipulasi oleh proses aplikasi. Kami mempertimbangkan tiga jenis proses C (komputasi), R (baca), dan W (tulis). Dalam proses C, hanya perhitungan yang dilakukan seperti yang dibahas dalam aplikasi CP. Artinya, sumber daya CPU sebagian besar dikonsumsi. Dalam proses R dan W, file dalam drive penyimpanan seperti hard disk drive (HDD) masing-masing dibaca dan ditulis. Kami menentukan model konsumsi daya server tempat proses membaca dan menulis file dalam jenis drive penyimpanan. Di sini, kami memperoleh model konsumsi daya dari server penyimpanan yang serupa dengan model konsumsi daya dua-status sederhana. Artinya, server mengkonsumsi daya maksimum jika setidaknya satu proses dilakukan.

3.2 STUDI TERKAIT

Dalam teknologi TI hijau, total konsumsi daya listrik dari sistem informasi harus dikurangi. Berbagai teknologi perangkat keras seperti CPU konsumsi daya rendah dikembangkan. Misalnya, AMD memproduksi prosesor Opteron di mana efisiensi daya ditingkatkan. Intel memproduksi prosesor Xeon seri 5600 dimana konsumsi daya dapat diatur secara otomatis sehingga kinerja server disesuaikan dengan trafik.

Dalam jaringan sensor nirkabel dan jaringan ad hoc seluler, algoritma perutean untuk mengurangi konsumsi daya setiap node dibahas. Secara khusus, dibahas bagaimana menggunakan node sensor untuk mengurangi konsumsi daya baterai setiap node sensor.

Sebuah sistem komputasi awan terdiri dari sejumlah besar komputer server seperti sistem file Google. Berbagai jenis algoritma *load balancing* dibahas untuk sistem terdistribusi. Bevilacqua membahas metode untuk mendapatkan *load balancing* yang efisien untuk aplikasi paralel data melalui penetapan data dinamis dan mekanisme prioritas sederhana pada sistem cluster heterogen. Penelitian yang dilakukan oleh Coljanni membahas algoritma load balancing yang menyesuaikan nilai *time-to-live* dari penjadwal berbasis DNS untuk server Web heterogen yang didistribusikan secara geografis. Linux Virtual Server (LVS) adalah perangkat lunak untuk membangun sistem cluster dan beberapa jenis algoritma load balancing diimplementasikan seperti algoritma *round-robin* dan *least-connection*. Dengan algoritma ini, perhitungan dimaksimalkan tetapi konsumsi daya tidak diperhitungkan.

Jika tidak ada permintaan di server, status komponen server diubah ke mode tidur nyenyak dan setiap aktivitas ditangguhkan hingga permintaan baru datang. Namun, sulit untuk mengaktifkan dan menonaktifkan server secara dinamis di beberapa jenis sistem seperti sistem kritis misi. Pendekatan ini dapat diadopsi hanya untuk sistem cluster di mana setiap server dikelola secara terpusat. Selain itu, komputer tidak dapat dimatikan oleh orang lain yang berbeda dari pemiliknya dalam jenis model komputasi lain seperti *peer-to-peer* (P2P) dan model grid. Dalam bab ini, kami mempertimbangkan sistem terdistribusi seperti sistem P2P di mana tidak ada koordinasi terpusat. Kami tidak mempertimbangkan pendekatan mematikan

server untuk mengurangi konsumsi daya total server. Kami lebih suka mengambil pendekatan untuk memilih server hemat energi dalam jaringan.

Ada berbagai jenis produk disk dengan spesifikasi yang berbeda seperti solid state drive (SSD) dan hard disk drive (HDD). Carrera et al. membahas empat pendekatan untuk mengurangi energi disk dengan menggunakan produk disk yang berbeda di server jaringan. Di kertas, konsumsi energi dapat dikurangi tanpa penurunan kinerja server jaringan di multi-speed disk. Namun, disk multi-kecepatan masih bukan produk umum. Dalam bab ini, kami mempertimbangkan drive SSD dan HDD umum. Kami mempertimbangkan tiga jenis proses aplikasi C, R, dan W yang memanipulasi file di drive penyimpanan. Kemudian, kami menentukan model konsumsi daya server tempat proses membaca dan menulis file dalam jenis drive penyimpanan. Berdasarkan model konsumsi daya server penyimpanan, kami membahas cara memilih server dalam kumpulan server penyimpanan untuk setiap permintaan dari klien.

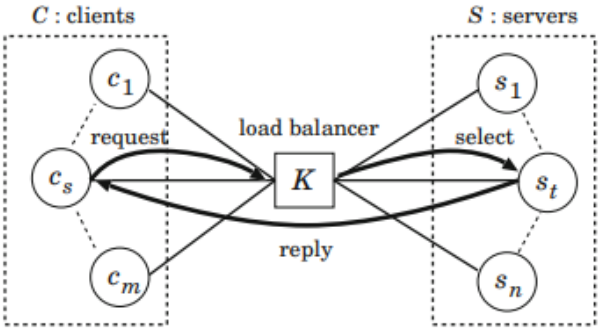
Karakteristik penggunaan daya agregat dari koleksi besar server untuk tiga jenis aplikasi skala besar. CPU dan memori mendominasi total konsumsi daya server dan model yang diusulkan terutama menggunakan pemanfaatan CPU untuk memperkirakan total konsumsi daya server. Kemudian, strategi untuk memaksimalkan jumlah peralatan komputasi yang dapat digunakan di pusat data dengan kapasitas daya yang diberikan diusulkan dengan menggunakan penskalaan frekuensi-tegangan CPU dan perubahan konsumsi daya yang tidak digunakan.

Dalam bab ini, kami mengklasifikasikan aplikasi ke dalam jenis aplikasi CP, CM, dan ST. Kami mempertimbangkan konsumsi daya total server untuk melakukan tiga jenis proses aplikasi. Kami tidak mempertimbangkan konsumsi daya setiap komponen server untuk mengurangi konsumsi daya total server. Dalam satu pendekatan, kami ingin mengusulkan model tingkat makro untuk menunjukkan konsumsi daya server untuk melakukan proses aplikasi. Pertama, kami mengukur konsumsi daya server dan kemudian memperjelas parameter apa yang paling mendominasi konsumsi daya server.

3.3 MODEL SISTEM

Server dan Klien

Misalkan S adalah sekumpulan beberapa server s_1, ..., s_n ($n \geq 1$) yang masing-masing mendukung klien dengan layanan yang sama seperti yang ditunjukkan pada Gambar 3.1. Misalnya, setiap server menyimpan replika objek data d di aplikasi CM. Misalkan C adalah kumpulan klien c_1, ..., c_m ($m \geq 1$) yang mengeluarkan permintaan ke server di kumpulan server S untuk mendapatkan layanan. Server dan klien saling berhubungan dalam jaringan. Kami menganggap jaringan dapat diandalkan dan sinkron. Artinya, tidak ada pesan yang hilang dan setiap pesan dikirimkan ke proses tujuan dalam waktu yang konstan di dalam jaringan.



Gambar 3.1. Model sistem

Klien cs pertama-tama mengeluarkan permintaan ke load balancer K seperti yang ditunjukkan pada Gambar 3.1. Penyeimbang muatan K memilih satu st server di set server S dan meneruskan permintaan ke st server. Di sini, server menghabiskan komputasi, komunikasi, dan sumber daya penyimpanan untuk melakukan permintaan. Misalnya, sumber daya CPU digunakan untuk menyandikan data multimedia dan melakukan perhitungan ilmiah dalam aplikasi CP. Dalam aplikasi penyimpanan (ST), drive penyimpanan dimanipulasi. Saat menerima permintaan, server st membuat proses pi. Proses pi dilakukan pada server st. Di sini, istilah proses berarti proses aplikasi di server untuk menangani permintaan yang dikeluarkan oleh klien di bab ini. Suatu proses yang mengkonsumsi sumber daya CPU dengan komputasi disebut sebagai proses komputasi (CP). Suatu proses yang mengkonsumsi sumber daya komunikasi dengan mentransmisikan data disebut sebagai proses komunikasi (CM). Suatu proses yang membaca dan menulis file dalam drive penyimpanan disebut sebagai proses penyimpanan (ST).

Proses di Server

Proses dilakukan pada server st. Sebuah proses yang dilakukan pada server st saat ini pada waktu τ. Sebuah proses yang telah berakhir sebelum waktu τ sebelumnya pada waktu τ. Biarkan *P* menjadi satu set proses aplikasi *p1, ..., pl (l ≥ 1)* yang akan dilakukan pada server di set server *S.*

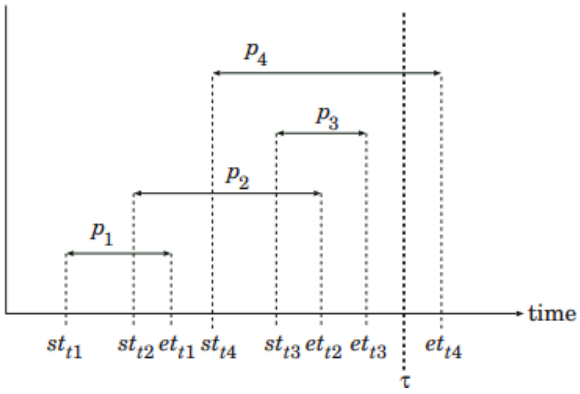
Biarkan *CPt(τ)* menjadi satu set proses saat ini di st server pada waktu τ. *Nt(τ)* menunjukkan *jumlah CPt(τ)* dari proses saat ini di server st. Jika *CPt(τ) = pi* , proses pi secara eksklusif dilakukan pada server st pada waktu τ. Artinya, hanya satu proses pi yang dilakukan dan tidak ada proses lain di server st pada waktu τ. Jika *pi ∈ CPt(τ)* dan *Nt(τ) > 1*, proses *pi* disisipkan dengan proses lain *pj ∈CPt(τ) (i ≠ j)* pada waktu τ. Artinya, banyak proses dilakukan secara bersamaan pada waktu τ. Biarkan *CP(τ)* menunjukkan serangkaian proses saat ini pada setiap server di server yang ditetapkan S pada waktu τ, *CP(τ) = pt∈S CPt(τ).*

Setiap proses *pi* dimulai pada server *st* pada waktu stti dan berakhir pada waktu *etti*. Sebuah proses *pi* mendahului proses lain *p j* di server *st (pi → t p j)* if *etti < sttj*. Sebuah proses *pi* diselingi dengan proses lain *p j (pi |t pj)* if *ettj ≥ etti ≥ stti* atau *ettj ≥ stti ≥ sttj*. Sebuah proses pi terhubung dengan proses lain *pj (pi ↔t pj*) jika (1) pi disisipkan dengan *pj (pi |t pj)* atau (2) *pi* disisipkan dengan beberapa proses *pk* yang dihubungkan dengan pj (*pi |t pk* dan *pk ↔ pj*). Jadwal proses *p1, ..., pm* menunjukkan riwayat eksekusi proses pada server *st*. Jadwal

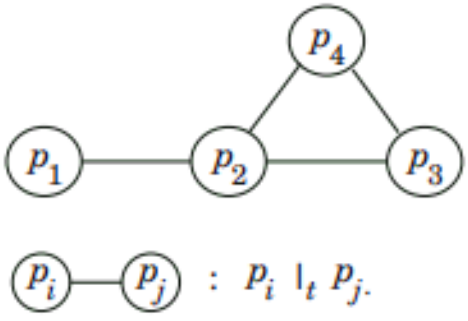
scht adalah sekumpulan proses yang dipesan sebagian yang dilakukan pada server *st* dalam relasi preseden *t* dan dalam relasi terhubung *t.*

Simpul *Kt(pi)* adalah subset penutup dari proses dalam *scht* jadwal yang terhubung dengan proses *pi,* yaitu *Kt(pi)* = {*pj* | *pj* $\leftrightarrow_t$ *pi*}. Jadwal *scht* dibagi menjadi simpul yang terpisah berpasangan *Kt1, ..., Ktlt.* Misalkan pj dan pk adalah sepasang proses dalam simpul *Kt(pi)* dimana *pj* pertama dimulai pada waktu *sttj* dan *pk* terakhir diakhiri pada waktu *ettk.* Waktu eksekusi *T Kt* dari simpul *Kt(pi)* adalah *ettk - sttj*. Misalkan *KPt(τ)* adalah simpul arus yang merupakan kumpulan proses saat ini atau sebelumnya yang terhubung dengan setidaknya satu proses saat ini dalam himpunan *CPt(τ)* pada waktu τ.

Gambar 3.2 menunjukkan scht jadwal dari empat proses p_1, p_2, p_3, dan p_4 pada server *st*. Di sini, proses *p1* diselingi dengan proses *p2 (p1* $|_t$ *p2).* Selain itu, *p2* $|_t$ *p3*, *p2* $|_t$ *p4*, dan *p3* $|_t$ *p4*. Gambar 3.3 menunjukkan bagaimana proses saling terkait satu sama lain. Di sini, sebuah node menunjukkan proses dan sisi antara node *pi* dan *pj* menunjukkan hubungan interleaved *pi* $|_t$ *pj*. Proses *p1* terhubung dengan proses p_2, p_3, dan p_4 (*p1* $|_t$ *p2, p1* $|_t$ *p3*, dan *p1* $|_t$ *p4*). Dengan demikian, semua proses saling terhubung satu sama lain. Di sini, simpul Kt(p1) adalah himpunan p_1, p_2, p_3, p_4 dari proses. $Kt(p_1) = Kt(p_2) = Kt(p_3) = Kt(p_4)$. Misalkan hanya proses p4 yang dilakukan pada server st pada waktu τ seperti yang ditunjukkan pada Gambar 3.2. *CPt(τ)* = p_4 . Sebuah proses *p1* pertama dimulai pada waktu *stt1* dan proses p_4 terakhir berakhir pada waktu ett_4 di simpul $Kt(p_4)$. Maka waktu eksekusi *T Kt* dari simpul $Kt(p_4)$ adalah ett_4 - *stt1*. Tidak ada proses yang dilakukan sebelum waktu *stt1* dan setelah waktu *ett4* sementara setidaknya satu proses dilakukan antara waktu *stt1* dan stt_4.



Gambar 3.2. Simpul



Gambar 3.3. Relasi interleave

3.4 EKSPERIMEN

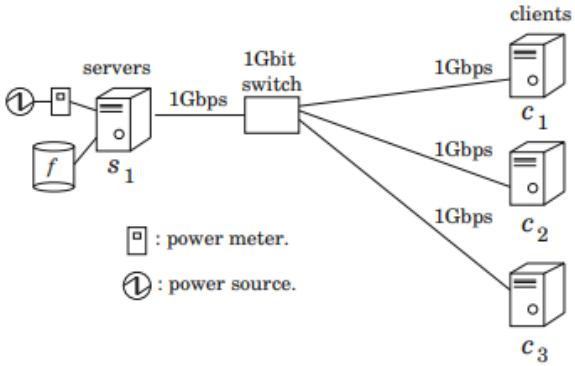
Aplikasi CP

Lingkungan Eksperimental

Kami pertama kali mengukur konsumsi daya server *s1* untuk melakukan proses aplikasi CP. Gambar 3.4 menunjukkan lingkungan percobaan. Server *s1* saling terhubung dengan tiga klien *c1, c2,* dan *c3* dalam jaringan satu-Gbps. Server *s1* menampung data d yang berukuran panjang 544 Mega byte [MB]. Tabel 3.1 menunjukkan spesifikasi server *s1*. Setiap klien *cs* mengeluarkan permintaan ke server *s1*. Setelah menerima permintaan, server s1 mengkompresi data *d* menjadi file *f* dengan menggunakan format gzip.

Tabel 3.1. Server *s1*

Server	*s1*
Jumlah CPU	1
Jumlah core/CPU	1
CPU	AMD Athlon 1648B (2.7GHz)
Penyimpanan	4.096 MB
Disk	150GB, 7200rpm
NIC	Broadcom Gbit Ether (1Gbps)



Gambar 3.4. Lingkungan eksperimental

Eksekusi Permintaan Secara Bersamaan

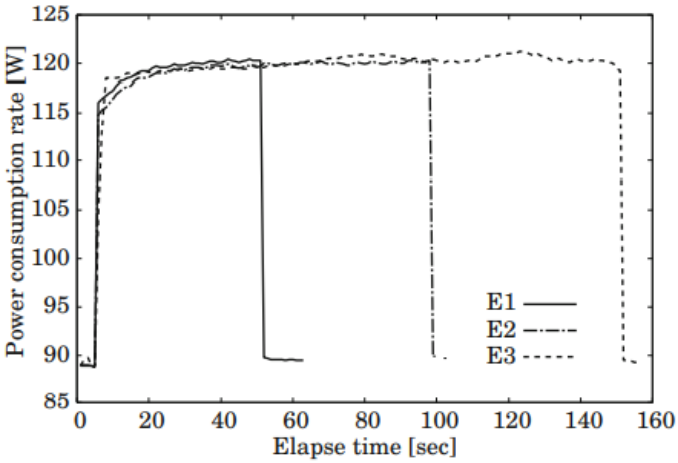
Kami mengukur berapa banyak daya listrik yang dikonsumsi server *s1* untuk melakukan tiga jenis eksperimen berikut dengan menggunakan pengukur daya UWMeter:

1. Klien *c1* mengeluarkan permintaan ke server *s1*.
2. Sepasang klien *c1* dan *c2* secara bersamaan mengeluarkan permintaan ke server *s1*.
3. Tiga klien *c1, c2*, dan *c3* secara bersamaan mengeluarkan permintaan ke server *s1*.

Gambar 3.5 menunjukkan tingkat konsumsi daya [W] untuk waktu percobaan [detik]. Tabel 3.2 merangkum total waktu eksekusi dan tingkat konsumsi daya yang diperoleh dari hasil percobaan. Dibutuhkan 44 [detik], 93 [detik] dan 146 [detik] untuk semua proses bersamaan untuk mengompresi data dalam eksperimen 1, 2, dan 3, secara berurutan. Diperlukan waktu 2,1 kali dan 3,3 kali lebih lama untuk mengompres data pada eksperimen 2 dan 3, masing-masing, daripada eksperimen 1. Setiap proses dilakukan di server dengan frekuensi clock

maksimum. Semakin banyak jumlah proses yang dilakukan secara bersamaan, semakin besar sumber daya komputasi yang dikonsumsi dan semakin lama waktu yang dibutuhkan untuk melakukan semua proses bersamaan karena peralihan proses seperti overhead.

Server *s1* mengkonsumsi daya listrik untuk mengompres data dengan kecepatan 119 [W], 120 [W], dan 121 [W] masing-masing dalam eksperimen 1, 2, dan 3. Di sini, jika setidaknya satu proses kompresi dilakukan di server, daya listrik dikonsumsi secara maksimal di server *s1*.



Gambar 3.5. Eksekusi permintaan secara bersamaan

Tabel 3.2. Eksekusi permintaan secara bersamaan

Eksperimen	Waktu [detik]	Tingkat konsumsi daya [W]
1	44	119
2	93	120
3	146	121

Aplikasi CM

Lingkungan Eksperimental

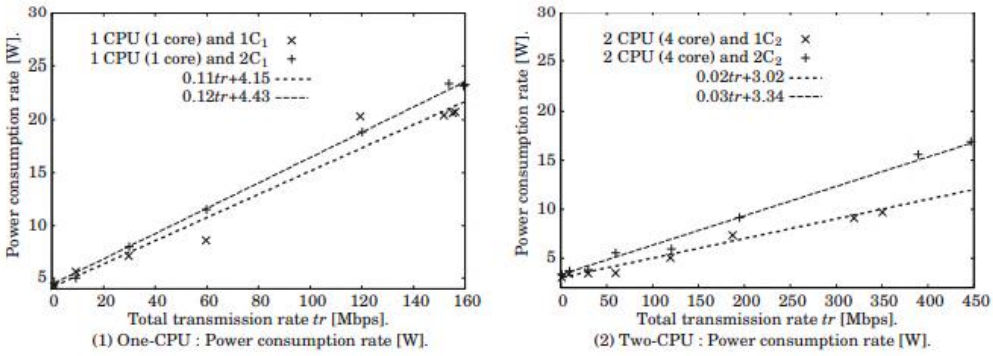
Kami mengukur berapa banyak daya listrik yang dihabiskan komputer untuk mengirimkan file ke komputer lain. Seperti yang ditunjukkan pada Gambar 3.6, sepasang server *s1* dan *s2* saling terhubung dengan sepasang klien *c1* dan *c2* dalam jaringan 1Gbps. Server *s1* dilengkapi dengan CPU satu inti seperti yang ditunjukkan pada Tabel 3.3. Server *s2* terdiri dari sepasang CPU dua inti. Artinya, bandwidth *bts* dari setiap *st* server ke setiap client *cs* adalah 1Gbps (t = 1, 2). Setiap klien *cs* mengunduh file f dari salah satu server. Ukuran file f panjangnya 43.051.806 byte. Di sini, kami mengukur konsumsi daya total server s1 dan s2.

Untuk setiap st server, kami mempertimbangkan dua jenis eksperimen, lingkungan satu klien *(1Ct)* dan dua klien (2Ct) (t = 1, 2). Di lingkungan *1Ct*, satu klien, misalnya *c1* mengunduh file *f* dari server *st*. Di lingkungan *2Ct,* sepasang klien *c1* dan *c2* secara bersamaan mengunduh file *f* dari server *st.*

Hasil Percobaan

Server st mengkonsumsi daya listrik untuk mengirimkan file ke klien. Di lingkungan *1C1*, server *s1* mentransmisikan file *f* ke satu klien, misalkan *c1* dengan kecepatan *tr11*. Di sini,

server *s1* terdiri dari satu CPU satu inti. Tingkat transmisi maksimum *Maxtr1* adalah 160 [Mbps] di jaringan bandwidth *b11* = 1G [bps]. Di lingkungan *2C1*, server *s1* secara bersamaan mentransmisikan file *f* ke sepasang klien *c1* dan *c2.* Di sini, *tr1 = tr11 + tr12*. Gambar 3.7 (1) menunjukkan tingkat konsumsi daya server *s1* untuk total tingkat transmisi tr1. Pada tingkat yang lebih tinggi *tr1* server *s1* mentransmisikan file *f*, semakin besar jumlah daya yang dikonsumsi server *s1.* Kami memperoleh rumus perkiraan *PC1(tr)* untuk menunjukkan laju konsumsi daya server *s1* untuk laju transmisi total *tr* [Mbps] dengan menerapkan metode kuadrat-terkecil pada hasil percobaan.



Gambar 3.7. Tingkat konsumsi daya [W]

Pada Gambar 3.7 (1), garis putus-putus tebal menunjukkan konsumsi daya yang diperkirakan dari server *s1* di mana satu klien mendownload file *f* dari server *s1*. Garis putus-putus menunjukkan perkiraan konsumsi daya server *s1* di mana sepasang klien *c1* dan *c2* secara bersamaan mengunduh file *f* dari server *s1*. Biarkan *PC1(tr)* dan *PC2(tr)* menjadi tingkat konsumsi daya server *s1* di *1C1* dan *2C1*, masing-masing, pada laju transmisi total *tr*. Dalam server CPU tunggal *s1,* laju konsumsi daya *PC1(tr)* sebanding dengan total laju transmisi *tr.*

1C1 : *PC1(tr)* = 0.11*tr* + *4.15* [W]
2C1 : *PC2(tr)* = 0.12*tr* + *4.43* [W]

Selanjutnya, kami mempertimbangkan server lain *s2* dengan sepasang CPU dua inti. Di sini, kecepatan transmisi maksimum *Maxtr2* dari server *s2* adalah 450 [Mbps]. Kami mengukur tingkat konsumsi daya untuk total tingkat transmisi *tr2* untuk lingkungan *1C2* dan *2C2*. Gambar 3.7 (2) menunjukkan laju konsumsi daya [W/detik] server *s2* untuk total laju transmisi tr. Gambar 3.7 (2) menunjukkan bahwa semakin tinggi kecepatan transmisi server *s2*, semakin besar konsumsi daya yang dikonsumsi *s2*. Formula perkiraan *PC1(tr)* dan PC2(t*r)* dari laju konsumsi daya server *s2* untuk laju transmisi total *tr* [Mbps] diberikan di lingkungan *1C2* dan *2C2*:

1C1 : *PC1(tr)* = 0.02*tr* + *3.02* [W]
2C1 : *PC2(tr)* = 0.03*tr* + *3.34* [W]

Tingkat peningkatan konsumsi daya server *s2* di lingkungan *2C2* sekitar 1,5 kali lebih besar dari lingkungan *1C2*. Dibandingkan dengan lingkungan satu-CPU *1Ct,* tingkat konsumsi daya tidak terlalu meningkat untuk peningkatan laju transmisi di lingkungan dua-CPU *2Ct.*

Mengikuti percobaan, tingkat konsumsi daya *PCt(tr)* dari *st* server meningkat secara linier untuk tingkat transmisi *trt(τ) (0 ≤ trt(τ) ≤ Maxtrt)* sebagai berikut:

$$PC1(tr) = \beta t(m) \cdot \delta t \cdot trf(\tau) + minEt \qquad (3.1)$$

Di sini, δt adalah konsumsi daya untuk mengirimkan satu Mbits [W/Mb] untuk lingkungan *1Ct*. δt tergantung pada jenis server st. Seperti yang ditunjukkan pada Gambar 3.7, semakin banyak jumlah klien, semakin besar jumlah daya listrik server yang dikonsumsi. $\beta t(m)$ menunjukkan berapa banyak konsumsi daya yang meningkat untuk jumlah m klien, $\beta t(m) \geq 1$ dan $\beta t(m) \geq \beta t(m - 1)$. Ada maxmt titik tetap sehingga $\beta t(maxmt - 1) \leq \beta t(maxmt) = \beta t(maxmt + h)$ untuk $h \geq 0$. minEt memberikan tingkat konsumsi daya minimum dari server st di mana tidak ada file yang ditransmisikan. $\beta t(maxmt)\ \delta t\ Maxtrt + minEt$ memberikan tingkat konsumsi daya maksimum *maxEt* dari server *st.*

Aplikasi ST

Lingkungan Eksperimental

Penulis menunjukkan hasil eksperimen pada konsumsi daya server untuk melakukan proses dalam aplikasi ST. Kami pertama-tama mengukur berapa banyak daya listrik yang dikonsumsi server untuk melakukan proses dalam aplikasi ST. Di sini, proses membuat akses ke drive penyimpanan sekunder. Kami memperjelas faktor apa yang mendominasi konsumsi daya server berdasarkan hasil eksperimen.

Proses membaca dan menulis file di *hard disk drive* (HDD) dan *solid state drive* (SSD) dilakukan di server dengan Linux (Cent OS). Kami mengukur tingkat konsumsi daya [W] server dengan menggunakan pengukur daya listrik Metaprotocol UWMeter setiap seratus milidetik [J/0,1 detik]. Kami menganggap server st dengan CPU (Intel®Core™i7) dan memori 500 GB seperti yang ditunjukkan pada Tabel 3.4 di mana jenis proses dilakukan.

Tabel 3.4. Server dan drive

Server	CPU : Intel(R) Core(TM) i7 Memory : 6.00GB DDR3 3 x 2048 OS : Cent OS 64 bit
HDD	Capacity : 500GB Rotation : 7,200 rpm Read speed : 121.45MB/s Write speed : 106.27MB/s Interface : Serial ATA300
SSD	Capacity : 120GB Read Speed : 197.02MB/s Write Speed : 93.89MB/s Interface : Serial ATA300

Kami mempertimbangkan tiga jenis proses yang akan dilakukan pada server *st,* proses **Komputasi** (C), **Read** (R), dan **Write** (W). Proses R dan W masing-masing hanya membaca dan

menulis file. Dalam proses C, hanya sumber daya CPU yang dikonsumsi seperti yang dibahas dalam aplikasi CP. Proses R dan W membaca dan menulis data sebesar satu [GB] di drive penyimpanan dengan menggunakan panggilan sistem baca dan tulis. Jika beberapa proses R dan W dilakukan secara bersamaan di server, setiap pasangan proses *pi* dan *pj* mengakses file yang berbeda di drive penyimpanan yang sama. Konsumsi daya total server ditampilkan sebanding dengan ukuran data yang akan dibaca dan ditulis. Dalam satu panggilan sistem baca/tulis, unit data 1024[B] dibaca dan ditulis. Ukuran maksimum data yang dapat dibaca dan ditulis dalam satu system call adalah 1024 [B]. Model konsumsi daya sederhana diusulkan untuk melakukan proses C pada server. Jika setidaknya satu proses C dilakukan di server, server menghabiskan konsumsi daya maksimum. Server menghabiskan daya minimum jika tidak ada proses yang dilakukan, yaitu server dalam keadaan diam.

Kami mempertimbangkan jenis lingkungan berikut di mana jumlah total m $\geq$ (0) dari proses C, R, dan W dilakukan secara bersamaan di *st* server:

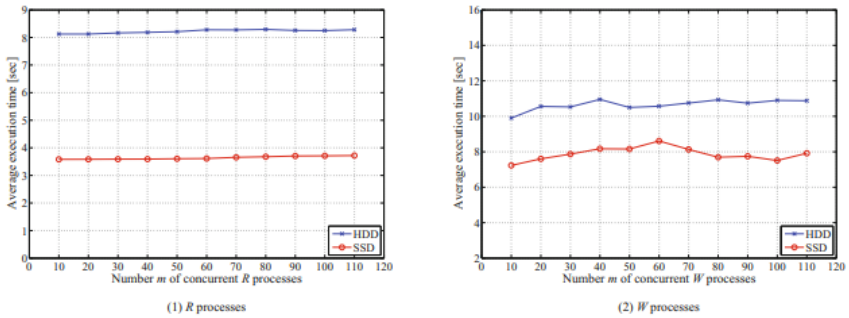
[Lingkungan]

I. *Ct* (m): Hanya proses C yang dilakukan secara bersamaan.
II. *Rt* (m): Hanya proses R yang dilakukan secara bersamaan.
III. *Wt* (m): Hanya proses W yang dilakukan secara bersamaan.
IV. *CRt* (m): Hanya proses R yang dilakukan bersamaan dengan proses C.
V. *CWt* (m): Hanya proses W yang dilakukan bersamaan dengan proses C.
VI. *RWt* (m, w): Proses R dan W dilakukan secara bersamaan. Tetapi tidak ada proses C yang dilakukan. Di sini, tulis rasio w menunjukkan rasio jumlah proses W terhadap m.
VII. *CRWt* (m, w): Proses R dan W dilakukan bersamaan dengan proses C di mana w menunjukkan rasio proses W.

Di sini, *RWt(m, 0) = Rt(m)* dan *RWt(m, 1) = Wt(m*). *CRWt(m, 0) = CRt(m)* dan *CRWt (m, 1) = CWt(m)*. Sebagai contoh, *RWt*(10, 0.5) berarti bahwa lima proses R dan lima proses W dilakukan secara bersamaan di server *st.*

Hasil Percobaan

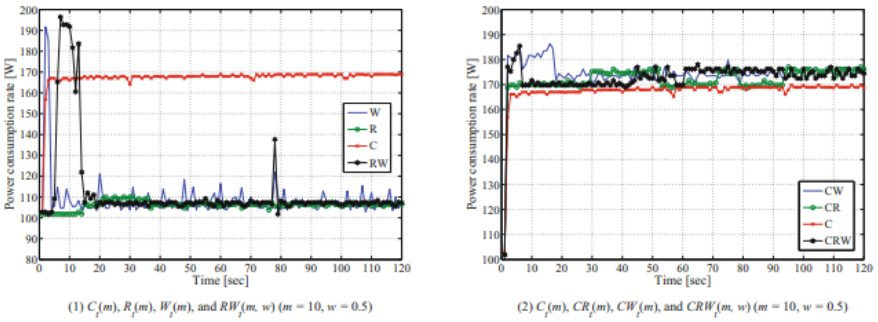
Pertama, kami mengukur waktu eksekusi rata-rata *AERt(m)* dan *AEWt(m)* [sec] dari setiap proses di lingkungan *Rt(m)* dan *Wt(m),* masing-masing, di mana jumlah *m* (0) proses dilakukan secara bersamaan pada server *st.* Gambar 3.8 menunjukkan waktu eksekusi *AERt(m)* dan *AEWt(m)* untuk m. Di sini, *AERt(m)* konstan untuk m 128 dan meningkat secara eksponensial untuk m 129.



Gambar 3.8. Waktu eksekusi

Dibutuhkan waktu lebih lama untuk menulis data daripada membaca, *AERt(m) < AEWt(m)*. Dalam hasil percobaan kami di *st* server, dibutuhkan waktu 30% lebih lama untuk menulis data di HDD daripada membaca, yaitu *AEWt(1) = 1.3 • AERt(1)* untuk HDD. *AEWt(1) = 1.1 • AERt(1)* untuk SSD.

Hasil eksperimen pada proses R dan W. Gambar 3.9 (1) menunjukkan tingkat konsumsi daya *etC(τ), etW(τ), etR(τ),* dan *etRW(τ)* [W] dari server st untuk melakukan proses di lingkungan *Ct(m), Wt(m), Rt(m),* dan *RWt(m, w)* masing-masing untuk *m* = 10 dan *w* = 0.5. Seperti yang ditunjukkan pada Gambar 3.9 (1), tingkat konsumsi daya tidak bergantung pada jumlah total m proses bersamaan. Selain itu, tingkat konsumsi daya *etA(τ)* dari server *st* maksimum jika setidaknya satu proses dilakukan pada waktu τ dalam lingkungan tipe A(C, R, W). Jalan server menggunakan minEt konsumsi daya minimum jika tidak ada proses yang dilakukan, yaitu jalan server dalam keadaan siaga. Menurut Gambar 3.9 (1), minEt tingkat konsumsi daya minimum adalah 101 [W] (10,1 [J/0,1 detik]). Biarkan maxCt, maxWt, dan maxRt menunjukkan tingkat konsumsi daya maksimum dari server st di mana hanya proses C, W, dan R yang dilakukan secara bersamaan. Biarkan Nt(τ) menjadi jumlah m proses yang dilakukan secara bersamaan di server st pada waktu τ. Laju konsumsi daya *etA(τ)* [W] dari server st diperoleh untuk setiap tipe lingkungan A C, R, W dari hasil percobaan yang ditunjukkan pada Gambar 3.9 sebagai berikut:



(1) $C_t(m)$, $R_t(m)$, $W_t(m)$, and $RW_t(m, w)$ ($m = 10$, $w = 0.5$) (2) $C_t(m)$, $CR_t(m)$, $CW_t(m)$, and $CRW_t(m, w)$ ($m = 10$, $w = 0.5$)

Gambar 3.9. Tingkat konsumsi daya

$$etc(\tau)\begin{cases} maxCt \\ minEt \end{cases}$$

$maxC_t$ jika setidaknya satu proses C dilakukan pada server *st* pada waktu τ di lingkungan C. $minE_t$ sebaliknya, yaitu $N_t(\tau) = 0$ dan tidak ada proses yang dilakukan pada *st.*

$$etW(\tau)\begin{cases} maxWt \\ minEt \end{cases}$$

maxWt jika setidaknya satu proses W dilakukan pada server st pada waktu τ di lingkungan W. *minEt* sebaliknya, *yaitu Nt(τ) = 0*.

$$etR(\tau)\begin{cases} maxRt \\ minEt \end{cases}$$

$maxR_t$ jika setidaknya satu proses R dilakukan pada server st pada waktu τ di lingkungan R. $minE_t$ sebaliknya, yaitu $N_t(\tau) = 0$.

$$etRW(\tau)\begin{cases} maxRWt \\ minEt \end{cases}$$

$maxRW_t$ jika setidaknya satu proses R dan setidaknya satu proses W dilakukan pada st pada waktu τ di lingkungan RW. $minE_t$ sebaliknya.

Di sini, $minE_t < maxW_t = maxR_t = maxRW_t = 108$ [W] $maxC_t = 168$ [W].

Tingkat konsumsi daya *etC(τ)* dari *st* server maksimum jika setidaknya satu proses C dilakukan pada st server pada waktu τ. Tingkat konsumsi daya *etC(τ)* minimum, yaitu *etC(τ) = minEt* jika tidak ada proses yang dilakukan pada server st pada waktu τ. Oleh karena itu, kami mempertimbangkan lingkungan *CRt(m), CWt(m),* dan *CRWt*(m, 0.5) di mana jumlah proses R dan W yang sama dilakukan pada server st secara bersamaan dengan proses C, masing-masing. Gambar 3.9 (2) menunjukkan tingkat konsumsi daya *etCR(τ), etCW (τ),* dan *etCRW (τ)* dari server *st* pada waktu τ di lingkungan *CRt(m), CWt(m), dan CRWt*(m, 0,5) untuk m = 10, masing-masing. Di sini, m = 10. Menurut Gambar 3.9 (2), *maxCRWt = maxCRt = maxCWt = 1.1 maxCt.* Di lingkungan *CRWt*(m, 0.5), jumlah daya yang dikonsumsi lebih besar daripada lingkungan *Ct(m). MinEt* tingkat konsumsi daya minimum dan tingkat konsumsi daya maksimum *maxRt, maxCt*, dan *maxCRWt* dari st server masing-masing adalah 101, 109, 168, dan 176 [W].

3.5 MODEL KONSUMSI DAYA

Aplikasi CP

Pada bagian ini, kami mendefinisikan sepasang model komputasi dan model konsumsi daya server untuk melakukan proses dalam aplikasi tipe komputasi (CP).

Model Perhitungan

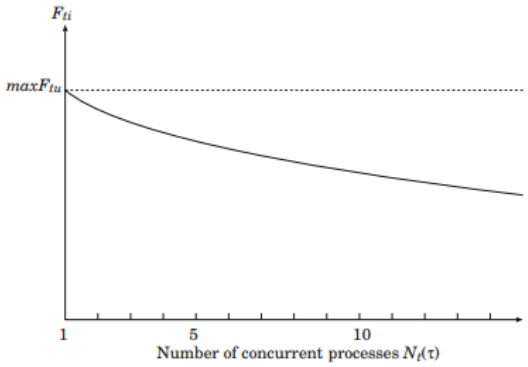
A. Laju Perhitungan Server

Proses yang menghabiskan sumber daya CPU dilakukan di server st. Pertama, kita membahas model komputasi server untuk melakukan proses. Kami mempertimbangkan parameter berikut untuk menentukan model komputasi st server:

- *Tti* = total waktu komputasi dari proses pi pada server st [detik].
- minTti = waktu komputasi minimum dari proses pi yang secara eksklusif dilakukan pada server *st (1 minTti Tti)* [detik].
- *maksTi = maks(minT1i, ..., minTni).*
- *minTi = min(minT1i, ..., minTni).*
- *Fti = 1 / Tti* = rata-rata Computing Rate (ACR) dari proses pi pada server st (0 < Fti 1 / menit 1) [1/dtk].

- *maxFti = 1 / minTti =* ACR maksimum dari proses pi pada server st.
- *maksFi = maks(maksF1i, ..., maksFni).*
- *minFi = min(maksF1i, ..., maksFni).*

Jika proses pi dilakukan secara eksklusif, yaitu hanya proses pi yang dilakukan tanpa proses lain, pada st server tercepat dan server paling lambat *su, minTti < minTui, minTi = minTti,* dan *maxTi = minTui*. Semakin banyak jumlah proses yang dilakukan, maka semakin lama pula waktu yang dibutuhkan untuk melakukan setiap proses pada sebuah server st. Biarkan αt(τ) menunjukkan laju degradasi komputasi dari server st pada waktu *τ (0 αt(τ) 1*) [1/detik] [Gambar 3.10]. *Di sini, αt(τ1) ≤ αt(τ2) ≤ 1 jika Nt(τ1) ≤ Nt(τ2)* untuk setiap pasangan waktu yang berbeda τ1 dan τ2. *αt(τ)* = 1 jika *Nt(τ) ≤ 1*. Dalam bab ini, αt(τ) diasumsikan sebagai *εNt (τ)−1* di mana 0 εt 1. Misalkan dibutuhkan 0,5 [detik] untuk melakukan proses pi secara eksklusif server Di sini, *Tti = minTti* = 0,5 [detik]. Di sini, *Fti = maxFti* = 1/0,5 = 2 [1/dtk]. Misalkan proses lain pu dimana *minTtu* = 0.5 dimulai pada waktu yang sama dengan proses *pi*. Jika *εt* = 0.98, *αt(τ) = εt* = 0.98 karena *Nt(τ)* = 2 untuk *τ ≥ st*. Oleh karena itu, *Ttu* = 1,02 *minTtu* = 0,51 [detik] dan *Ftu* = 1/0,51 = 1,96. Kami mendefinisikan tingkat komputasi yang dinormalisasi (NCR) *fti(τ)* dari proses *pi* pada server *st* sebagai berikut:



Gambar 3.10. Tingkat degradasi

[Tingkat Komputasi Normalisasi (NCR)]

$$fti(\tau) = \begin{cases} \alpha t(\tau) \cdot \dfrac{maxFti}{maxFi}[1/sec] \\ \alpha t(\tau) \cdot \dfrac{minTti}{minTi}[1/sec] \end{cases} \quad (3.2)$$

Maksimum NRC *maxfti* menunjukkan *maxFti / maxFi. 0 ≤ fti(τ) max fti ≤ 1. fti(τ)* menunjukkan berapa jumlah langkah proses p*i* yang dilakukan pada server *st* pada waktu *τ*. *max ft* didefinisikan sebagai NCR maksimum dari server st. Di sini, *max fti(τ) ≤ max ft* untuk setiap proses pi dan waktu τ. Selanjutnya, misalkan proses pi dimulai dan diakhiri pada server st pada waktu *stti* dan *etti*, masing-masing. Di sini, *Tti = etti - stti.*

$$\int_{sTti}^{eTti} fti(\tau)dt = minTi \cdot \int_{sTti}^{eTti} \frac{\alpha t(\tau)}{minTti} = minTi$$

(3.3)

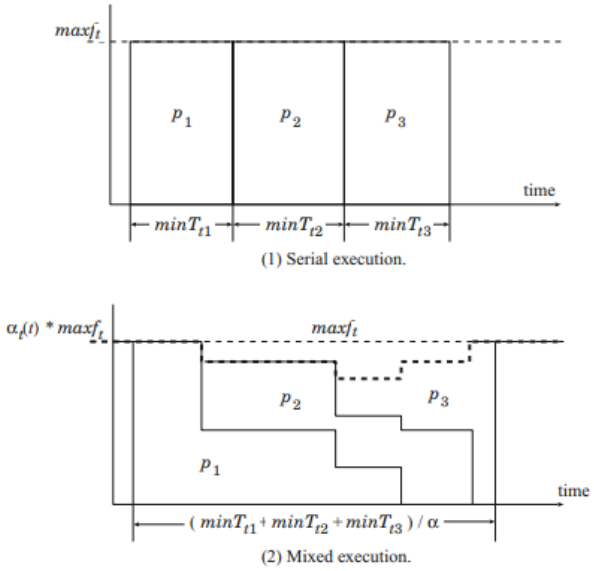
B. Model Perhitungan Sederhana

Dari hasil percobaan yang disajikan pada bagian sebelumnya, kami memperoleh model komputasi sederhana di mana setiap st server memenuhi properti berikut:

I. *max fti = max ftu = max ft* untuk setiap pasangan proses yang berbeda *pi* dan *pj.*
II. *Σpi∈Pt(τ) fti(τ) = αt(τ) • maks ft.*

Kondisi pertama menunjukkan bahwa setiap proses dilakukan pada server dengan frekuensi clock maksimum. Kondisi kedua berarti semakin banyak jumlah proses yang dilakukan secara bersamaan, semakin besar sumber daya komputasi yang digunakan. αt(τ) ≤ 1. Di sini, misalkan Kt menjadi simpul dalam scht jadwal server st, di mana st dan et adalah waktu mulai dan waktu penghentian, masing-masing. Waktu eksekusi T Kt dari simpul Kt adalah Σpi ∈ KtminTti / α dimana α = $\int_{st}^{et}$ αt(τ) dτ / (et − st).

Mari kita pertimbangkan tiga proses *p1, p2,* dan *p3* pada server st seperti yang ditunjukkan pada Gambar 3.11 (1). Pertama, misalkan proses dilakukan secara serial. Di sini, simpul *Kt* adalah *p1, p2, p3.* Waktu eksekusi *T Kt* dari simpul *Kt* adalah *ett3 - stt1 = minTt1 + minTt2 + minTt3* karena *αt(τ) = 1* untuk *stt1 τ ett3*. Selanjutnya, tiga proses *p1, p2, dan p3* secara bersamaan dilakukan di server st seperti yang ditunjukkan pada Gambar 3.11 (2). Di sini, simpul *Kt* adalah *{p1, p2, p3}*. Proses p1 pertama dimulai pada waktu *stt1*. Pada saat *stt2,* proses *p2* dimulai. Di sini, *αt(τ) = ε2−1 = εt* untuk *stt2 ≤ τ ≤ stt3*. Kemudian, pada saat stt3, proses *p3* dimulai. Di sini, αt(τ) = ε3−1 = ε2 untuk stt3 ≤ τ ett1. Di sini, *T Kt = ett3 - stt1 = (minTt1 + minTt2 + minTt3) / α. α = [(stt2 - stt1) + (stt3 - stt2)εt + (ett1 - stt3)ε2 + (ett2 - ett1)εt + (ett3 - ett2)] / (ett3 - stt1).*



Gambar 3.11. Waktu eksekusi simpul saat ini

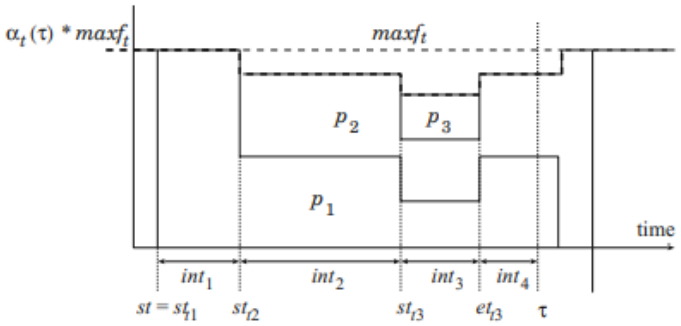
Itu tergantung pada algoritma penjadwalan berapa banyak *NCR fti(τ)* dari setiap pi proses saat ini pada waktu τ. Dalam bab ini, kami menganggap sumber daya CPU dialokasikan secara merata ke setiap proses saat ini sebagai berikut:

$$fti(\tau) = \alpha t(\tau) \cdot maxft. |CPt(\tau)|$$

(3.4)

C. Perkiraan Waktu Pengakhiran

Kami membahas bagaimana memperkirakan waktu penghentian simpul saat ini *KPt(τ) = {pt1, ..., ptlt* dari proses di server st, di mana waktu mulainya adalah *st.* Misalkan proses *pi* dimulai pada server *st* pada waktu *τ,* yaitu *stti = τ*. Biarkan *EVt(τ)* menjadi himpunan waktu mulai dan waktu terminasi dari proses yang memulai atau mengakhiri dalam simpul saat ini *KPt(τ)* antara waktu *st* dan waktu saat ini *τ* pada server *st.* Di sini, setiap elemen et dalam himpunan *EVt(τ)* disebut sebagai kejadian pada simpul *KPt(τ).* Acara di *EVt(τ)* diurutkan dalam urutan menaik. *EVt(τ) = e1, ..., el* dimana e1 el. Sebagai contoh, *EVt(τ) = stt1, stt2, stt3, ett3, τ* pada Gambar 3.12. Jika tidak ada simpul arus *KPt(τ)* pada server st pada waktu *τ*, proses *pi* adalah proses pertama pada simpul arus baru *KPt(τ)* dan *EVt(τ) = stt1* dimana *stt1 = τ*. Misalkan intk adalah interval waktu antara kejadian ek dan kejadian berikutnya *ek+1* dalam *EVt(τ),* yaitu *intk = ek+1 - ek (1 ≤ k≤ l -1 dan l ≥ 2). int1* = 0 jika l = 1. Biarkan *ACtik* menjadi jumlah perhitungan proses *pi* yang dilakukan pada *st* server selama intk interval. *Nt(intk)* menunjukkan jumlah proses yang dilakukan dalam interval intk. *ACtik* diberikan sebagai berikut:



Gambar 3.12. Peristiwa dalam simpul saat ini

$$ACtik = \frac{\left(et^{Nt(itk)-1} \cdot maxft\right)}{Nt(intk) \cdot intk}$$

(3.5)

Sebagai contoh, jumlah *Nt(int2)* dari proses konkuren selama selang waktu *int2* adalah 2 pada Gambar 3.12. Kemudian, *ACt12* dari proses *p1* dalam selang waktu *int2* di server *st* adalah *(εt•max ft) / 2•int2.*

Misalkan *INTt(τ)* adalah himpunan interval waktu *int1, ..., intl−1* antara waktu *st* dan waktu τ pada simpul saat ini *KPt(τ).* Jumlah total perhitungan *T ACti(τ)* yang dilakukan pada server st antara waktu st dan τ untuk setiap proses *pi* dalam *KPt(τ)* dihitung dengan prosedur berikut:

AmountOfPerformed(INTt(τ), KPt(τ)) {
T ACT = φ;
untuk setiap proses pi **dalam** *KPt(τ)*
{
T ACTi = 0;
untuk setiap interval waktu *intk* **dalam** *INTt(τ)*
{
if *pi ∈ Nt(intk), T ACti = T ACti + ACtik;*
}
T AC = TAC + {TACti};
}
Return*(T ACt);*
}

Dalam prosedur **AmountOfPerformed**(), jumlah total *T ACti* dari komputasi yang dilakukan oleh setiap proses *pi* dalam *KPt(τ*) antara waktu *st* dan *τ* pada server *st* disimpan dalam set *T ACt.* Di sini, *T ACt = T ACt1, ..., T ACtlt*. Biarkan *T ACt[i]* menunjukkan *T ACti.*

Jumlah total perhitungan yang harus dilakukan untuk menghentikan proses pi adalah *maxft · minTti* pada server *st*. Kuantitas perhitungan residual *RCQti* menunjukkan berapa banyak perhitungan yang harus dikeluarkan untuk menghentikan proses *pi* pada *st* server setelah waktu *τ. RCQti* diberikan sebagai berikut:

$$RCQti = (maxft \cdot minTti) - TACti \quad (3.6)$$

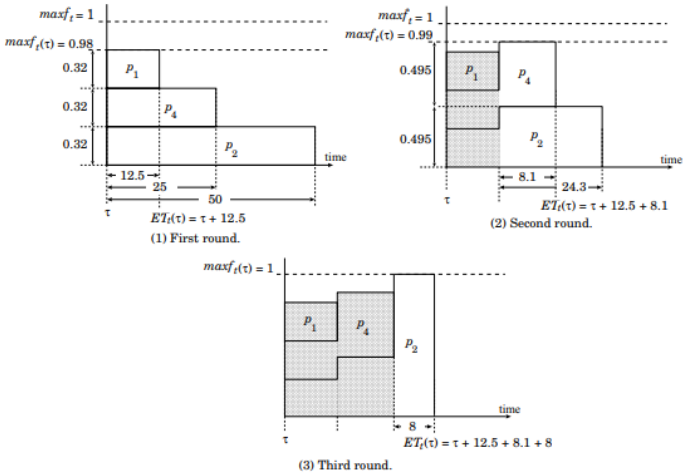
Sebuah tuple *(i, RCQti)* adalah sepasang pengenal *i* dari proses *pi* dan kuantitas komputasi residual *RCQti* dari proses pi pada waktu τ. *RCQti* dari setiap proses pi dalam *KPt(τ)* dapat dihitung sebagai berikut:

CalcRCQ*(INTt(τ), KPt(τ)) RCQt = φ;*
T ACt = **AmountOfPerformed**(*INTt(τ), KPt(τ));*
For every process pi **in** KPt(τ)
{
T ACti = T ACt[i]; / T ACti dari pi di T ACt; */*
RCQti = (max ft • minTti) - T ACti;
If *RCQti > 0, RCQt = RCQt ∪{(i, RCQti)};*

```
/* if RCQti ≤ 0, pi has terminated before time τ. */
}
Return(RC Qt);
}
```

Dalam prosedur **CalcRCQ(),** setiap pasang identifier i dan kuantitas perhitungan sisa *RCQti* dari proses *pi* yang harus dilakukan setelah waktu τ disimpan dalam *RCQt.* Biarkan *ETt(τ)* menjadi perkiraan waktu penghentian simpul saat ini *KPt(τ)* di mana server *st* dipilih untuk proses *pj* pada waktu *τ. ETpj (τ)* menunjukkan estimasi waktu negara dari proses *pj* pada server *st* pada waktu τ. Sepasang *ETt(τ)* dan *ETpj (τ)* dihitung sebagai berikut:

```
EstimateTermination(INTt(τ), KPt(τ), p j)
RCQt = CalcRCQ(INTt(τ), KPt(τ));
RCQt = RCQt j, maks ft minTtj ; /*( j, RCQti) dari pj ditambahkan ke RCQt.*/
SORT(RCQt); /*RCQt diurutkan dalam urutan menaik dari RCQti.*/
ETt(τ) = τ;
while (RCQt ≠ φ) { /* (1) */
Nt(τ) = |RCQt|;
max ft(τ) = εNt(τ)−1 • max ft;
E stTt = φ;
for every element (i, RCQti) dalam RCQt
{
        E stTti = (max ft(τ)/Nt(τ))−1 • RCQti;
        E stTt = E stTt ∪ {(i, E stTti)};
}
minEstTti=MIN(EstTt);
ETt(τ) = ETt(τ) + minEstTti;
For every elemen (i, E stTti) in (EstTt)
{
        if EstTti = minEstTti,
        {
                if i = j, ETpj (τ) = ETt(τ); RCQt = RCQt - {(i, RCQti)};
        }
                else
                {
                        (i, RCQti) = search(i, RCQt);
                        newRCQti = RCQti - (minE stTti • maks ft(τ) / Nt(τ));
                        (i, RCQti) = (i, newRCQti);
                }
        }
 }
return((ETt(τ), ETp (τ)));
}
```



Gambar 3.13. Perkiraan waktu penghentian

Misalkan server *st* dipilih untuk proses *p4* pada waktu τ pada Gambar 3.13. Di sini, kami berasumsi bahwa *RCQt = {(p1, 4), (p4, 8), (p2, 16)}* sesuai dengan prosedur **EstimateTermination().** Selain itu, *maksft* = 1 dan *εt* = 0,99. Di sini, perulangan **while** dilakukan sebanyak tiga kali [lihat baris (1)]. Pada putaran pertama, *max ft(τ) = ε3−1 1 = 0.992 1 = 0.98, E stTt = {(p1, 12.5}, (p4, 25), (p2, 50), dan ETt(τ) = τ + 12,5*, masing-masing, seperti yang ditunjukkan pada Gambar 3.13 (1). Kemudian, *RCQt* diubah menjadi p4, 4 , p2, 12 . Pada putaran kedua, *max ft(τ) = ε2−1 1 = 0.99 1 = 0.99, EstTt = (p4, 8.1), (p2, 24.3)* dan *ETt(τ) = τ + 12.5 + 8 = τ + 20.6,* masing-masing, sebagai ditunjukkan pada Gambar 3.13 (2). Di sini, proses *p4* adalah proses baru di mana st server dipilih pada waktu *τ.* Oleh karena itu, *ETp4(τ) = τ + 20,6*. Kemudian, *RCQt* diubah menjadi *(p2, 8)*. Pada putaran ketiga, *max ft(τ) = ε0 1 = 1, E stTt = (p2, 8), dan ETt(τ) = τ + 12.5 + 8.1 + 8 = τ + 28.6*, berturut-turut, seperti yang ditunjukkan pada Gambar 3.13 (3). Oleh karena itu, estimasi waktu terminasi *ETt(τ)* dari simpul arus *KPt(τ)* adalah *τ + 28.6* dan estimasi waktu terminasi *ETp4 (τ)* dari proses *p4* adalah *τ + 20,6.*

Model Konsumsi Daya

Kami membahas model konsumsi daya server st untuk aplikasi CP. Di sini, kami mempertimbangkan parameter berikut:

- *Et(τ)* = laju konsumsi daya listrik server st pada waktu *τ [W] (t = 1, aˆ€¦,n).*
- *maxEt* = tingkat konsumsi daya listrik maksimum dari server *st.*
- *minEt* = tingkat konsumsi daya listrik minimum dari server *st.*
- *maksE = maks(maksE1, ..., maksEn).*
- *minE = min(minE1, ..., minEn).*

Kami mendefinisikan tingkat konsumsi daya yang dinormalisasi *(NPCR) et(τ)* dari server *st* pada waktu τ sebagai berikut:

$$et(\tau) = \frac{Et(\tau)}{maxE} (\leq 1) \quad (3.7)$$

Biarkan *minet* dan *maxet* masing-masing menunjukkan tingkat konsumsi daya minimum *minEt / maxE* dan tingkat konsumsi daya maksimum *maxEt / maxE* dari *st* server. Di

sini, *et(τ)* menunjukkan rasio berapa banyak server yang mengkonsumsi daya listrik ke server yang paling banyak mengkonsumsi daya listrik pada waktu *τ*. Jika *st* server tercepat menghabiskan daya listrik secara maksimal dengan frekuensi clock maksimum, *et(τ) = maxet = 1*. Kami mendefinisikan model konsumsi daya sederhana untuk *st* server sebagai berikut:

$$et(\tau) = \begin{cases} maxet, & if\ Nt() \geq 1 \\ minet, & otherwise \end{cases} \quad (3.8)$$

Ini berarti, jika setidaknya satu proses dilakukan di st server, daya listrik dikonsumsi secara maksimal di *st* server. Jika tidak ada proses yang dilakukan pada server *st, et(τ) = minet.* Konsumsi daya yang dinormalisasi *NPCt(τ1, τ2)* [Ws] dari server *st* dari waktu *τ1* ke waktu *τ2* diberikan sebagai berikut:

$$NPCt(\tau 1 \cdot \tau 2) = \int_{\tau 2}^{\tau 1} et(\tau) dt \quad (3.9)$$

Aplikasi CM

Pada bagian ini, kami membahas model transmisi dan konsumsi daya untuk aplikasi komunikasi (CM).

Model Transmisi

Pertama, biarkan *trts(τ)* menjadi tingkat transmisi dari server st untuk mengirimkan file *fs* ke klien *cs* pada waktu τ. Misalkan server *st* secara bersamaan mengirim file *f1, ..., fm* ke set *CTt(τ)* klien *c1, ..., cm* dengan kecepatan transmisi *trt1(τ), ..., trtm(τ) (m ≥ 1),* masing-masing, pada waktu τ. *bts* menunjukkan bandwidth maksimum [bps] antara *st* server dan klien *cs*. Biarkan *Maxtrt* menjadi laju transmisi maksimum [bps] dari server *st (≤ bts*). Di sini, tingkat transmisi total *trt(τ)* dari server *st* pada waktu diberikan sebagai *trt(τ) = trt1(τ) + …… + trtm(τ).* Di sini, *0 ≤ trt(τ) ≤ Maxtrt.*

Setiap klien *cs* menerima pesan dari server dengan tingkat penerimaan *rrs(τ)* pada waktu τ. Biarkan *Maxrrs* menunjukkan tingkat penerimaan maksimum klien *cs.* Kami menganggap setiap klien *cs* menerima file dari paling banyak satu server dengan tarif *Maxrrs (= rrs(τ)).* Server *st* mengalokasikan setiap klien *cs* dengan laju transmisi *trts(τ)* sehingga *trts(τ) Maxrrs* pada waktu *τ.*

Biarkan *TRts* menjadi total waktu transmisi file *fs* dari *st* server ke klien *cs*. Jika server mengirim file ke klien lain secara bersamaan dengan klien *cs,* waktu transmisi *TRts* meningkat. Biarkan *minTRts* menunjukkan waktu transmisi minimum *|fs|/mnt(Maxrrs, Maxtrt)* [dtk] file *fs* dari *st* server ke *cs* klien di mana *|fs|* menunjukkan ukuran [bit] file *fs*. *TRts ≥ minRts.* Misalkan server *st* memulai dan mengakhiri transmisi file *fs* ke klien *cs* masing-masing pada waktu *st* dan *et.* Di sini, = *|fs|* dan waktu transmisi *TRts* adalah *(et - st).* Jika server *st* hanya

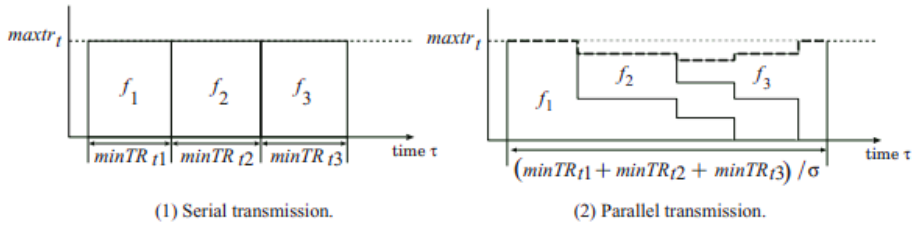
mengirim file *fs* ke klien *cs* pada waktu *τ, trts(τ) = min(Maxrrs, Maxtrt)* [bps]. Transmisi *et* τ server *st* masih harus mengirimkan ke klien *cs* pada waktu τ. Pertama, kami mempertimbangkan model di mana st server memenuhi properti berikut:

[**Model terikat server**] Jika *Maxrr1 + ••• + Maxrrm ≥ σt(τ) • Maxtrt, Σcs ∈CTt (τ) trts(τ) = σt(τ) • Maxtrt* untuk setiap waktu τ.

Di sini, *σt(τ) (≤ 1*) adalah faktor penurunan transmisi dari server st. Di sini, kita *asumsikan σt(τ) = γ1−|CTt (τ)| (0 < γ ≤ 1)* pada waktu τ. Semakin banyak jumlah klien yang ditransmisikan oleh server, semakin lama waktu yang dibutuhkan. Laju transmisi efektif dari server *st* adalah *σt(τ) Maxtrt.*

Misalkan server mengirim tiga file *f1, f2,* dan *f3* ke klien *c1, c2,* dan *c3,* masing-masing. Pertama, misalkan server *st* secara serial mengirimkan file *f1, f2,* dan *f3* ke klien *c1, c2*, dan *c3*, yaitu *ett1 = stt2* dan *ett2 = stt3* seperti yang ditunjukkan pada Gambar 3.14 (1). Di sini, waktu transmisi *TRt* adalah *ett3 - stt1 = minT Rt1 + minT Rt2 + minT Rt3*. Selanjutnya, misalkan server *st* pertama-tama mulai mentransmisikan file *f1* pada waktu *stt1* dan mengakhiri transmisi file *f3* pada waktu *ett3* seperti yang ditunjukkan pada Gambar 3.14 (2). Pada saat *stt2,* server *st* mulai mentransmisikan file *f2*. Di sini, *CTt(τ) = 2* dan *σt(τ) = γ1−2 = γ−1* untuk *stt2 ≤ τ ≤ stt3*. Kemudian, pada saat *stt3,* server *st* mulai mengirimkan file *f3*. Di sini, CTt(τ) = 3 dan σt(τ) = γ1−3 = γ−2 untuk stt3 ≤ τ ≤ ett1. Di sini, *TRt = ett3 - stt1 = (minT Rt1 + minT Rt2 + minT Rt3) / σ. σ = [(stt2 - stt1) + (stt3 - stt2)γ + (ett1 - stt3)γ2 + (ett2 - ett1)γ + (ett3 - ett2)] / (ett3 - stt1).* Di sisi lain, kami mempertimbangkan lingkungan lain di mana beberapa klien *cs* tidak dapat menerima file dari server dengan kecepatan transmisi maksimum *Maxtrt*, yaitu *Maxrrs < Maxtrt*. Oleh karena itu, laju transmisi *trts(τ)* adalah laju penerimaan maksimum *Maxrrs.*

[**Model terikat klien**] Jika *Maxrr1 + ••• + Maxrrm ≤ σt(τ) • Maxtrt, Σcs ∈CTt (τ) trts(τ) = Maxrr1 + ••• + Maxrrm* untuk setiap waktu *τ.*



(1) Serial transmission. (2) Parallel transmission.

Gambar 3.14. Waktu transmisi

Model Konsumsi Daya

Biarkan *PTt(τ)* menunjukkan tingkat konsumsi daya listrik [W] dari server st untuk mentransmisikan file ke klien pada waktu *τ (t = 1, ..., n). maxEt* dan *minEt* masing-masing menunjukkan konsumsi daya listrik maksimum dan minimum dari *st* server. Di sini, *minEt ≤ PTt(τ) ≤ maxEt. maxE* dan *min*E masing-masing menampilkan *maks(maxE1, ..., maxEn)* dan *min(minE1,..., minEn).* Di sini, kami berasumsi bahwa hanya aplikasi transfer file yang

dilakukan di setiap server *st*. Laju konsumsi daya listrik *PTt(τ)* dari server *st* pada waktu τ diberikan sebagai berikut:

$$PTt(\tau) = PCt\big(trt(\tau)\big) = \beta t(|CTt(\tau)| \cdot \delta t \cdot trt(\tau) + minEt)$$
(3.10)

Konsumsi daya *T PCt(τ1, τ2)* [Ws] dari server *st* dari waktu *τ1* ke waktu *τ2* diberikan sebagai berikut:

$$TPCt(\tau 1 \cdot \tau 2) = \int_{\tau 2}^{\tau 1} PTt(\tau) dt$$
(3.11)

Aplikasi ST

Pada bagian ini, kami membahas model untuk aplikasi penyimpanan (ST).

Model Akses

Dalam aplikasi ST, setiap proses pi dikarakterisasi berdasarkan tipe proses *tpi* ∈ {C, R, W} dan jumlah *li* [bit] komputasi dan akses. Jika proses *pi* adalah proses C*, li* adalah jumlah total komputasi dari *C* proses *pi.* Biarkan *ETti* menunjukkan waktu eksekusi untuk melakukan proses *pi* pada server *st.* Misalkan proses *C* pada *pi* dilakukan di server *st* tanpa proses lainnya. Di sini, dibutuhkan *METti* [detik] untuk melakukan proses *pi*. Semakin banyak jumlah proses yang dilakukan, semakin lama waktu yang dibutuhkan untuk melakukan setiap proses *pi,* yaitu *METti* ≤ *ETti*. Laju komputasi maksimum *maxFti* dari sebuah proses *pi* pada server *st* adalah 1/*METti* [1/sec]. Tercatat *maxFti = maxFtj* untuk setiap pasangan proses *pi* dan *pj. maxFt* menunjukkan tingkat perhitungan maksimum dari server *st,* yaitu *maxFt = maxFti* untuk setiap *pi* proses. Laju perhitungan *Fti(τ)* dari proses *pi* pada server *st* pada waktu *τ* menunjukkan berapa persen dari total perhitungan proses *pi* yang dilakukan pada waktu τ seperti yang dibahas pada subbab sebelumnya. Jika proses *pi* dilakukan tanpa proses apa pun di *st* server, *Fti(τ)* = 1/*METti* [1/sec] yang merupakan tingkat perhitungan maksimum *maxFti.* Laju komputasi maksimum *Ft(τ)* dari server *st* adalah *maxFti* untuk proses *pi, Ft(τ) = maxFt.* Dalam makalah ini, kami menganggap tingkat komputasi dialokasikan secara adil untuk setiap proses konkuren pada waktu τ, yaitu *Fti(τ) = αt · maxFt/|NCt(τ)|* untuk setiap proses *pi* dalam *NCt(τ).* Di sini, αt = ϵ dimana 0 < $\epsilon \leq 1$.

Dalam proses R atau W, *li* menunjukkan ukuran file yang akan dibaca atau ditulis. Proses R *pi* membaca file berukuran *si* sementara proses W *pj* menulis file berukuran l j [bit] di drive penyimpanan server *st*. Laju akses baca *ARti* dan laju akses tulis *AWtj* [bps] dari proses R dan W *pi* dan *pj* pada server *st* masing-masing diberikan sebagai *li/ETti* [bps] dan *lj/ETtj* [bps]. Pada hasil eksperimen yang ditunjukkan pada Gambar 3.8, laju baca *ARTi* dari proses *R pi* adalah 8,22 M [bps] dan laju tulis *AWsj* dari proses W *pj* adalah 10,65 M [bps] untuk HDD. Hal ini juga tersirat dari hasil eksperimen bahwa *ARti = ARtj* untuk setiap pasangan proses *R pi* dan *pj* dan *AWti = AWtj* untuk setiap pasangan proses *W pi* dan *pj* bahkan jika banyak proses dilakukan secara bersamaan di server *st. ARt* dan *AWt* masing-masing menunjukkan kecepatan baca dan tulis maksimum server untuk membaca dan menulis file. Biarkan *αtR(τ)*

dan *atW(τ)* menjadi laju baca dan tulis untuk proses R dan W untuk membaca dan menulis file pada server *st* pada waktu τ, berturut-turut.

Berdasarkan hasil eksperimen, kecepatan akses baca AtR(τ) dan kecepatan akses tulis padaW(τ) diperoleh sebagai berikut:

$$AtR(\tau)\begin{cases} ARt \\ 0 \end{cases}$$

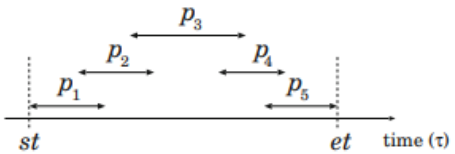
ARt jika setidaknya satu proses R dilakukan pada server st pada waktu τ.
0 sebaliknya.

$$AtW(\tau)\begin{cases} AWt \\ 0 \end{cases}$$

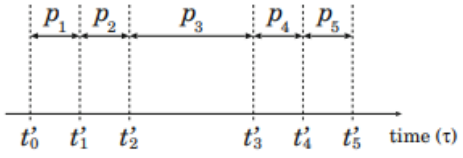
AWT jika setidaknya satu proses W dilakukan pada server st pada waktu τ.
0 sebaliknya.

Beberapa proses secara bersamaan dilakukan pada server *st.* Kami membahas berapa lama waktu yang dibutuhkan untuk melakukan proses pada server *st.* Biarkan *Kt* menjadi simpul proses *pt1, ptm* di server *st.* Waktu mulai *stt1* dari proses *p1* adalah minimum dan waktu akhir *ettm* adalah maksimum di simpul *Kt.* Artinya, tidak ada proses di simpul *Kt* yang dilakukan sebelum waktu *stt1* dan sesudah waktu *ettm.* Setidaknya satu proses dalam simpul *Kt* dilakukan kapan saja antara waktu *stt1* dan *ettm.* Waktu eksekusi *Tt* untuk melakukan setiap proses pada simpul Kt adalah ettm *stt1.*

Misalkan proses *p1, p2, p3, p4,* dan *p5* disisipkan, yaitu dilakukan secara bersamaan seperti yang ditunjukkan pada Gambar 3.15. Di sini, simpul *Kt(p1) = p1, p2, p3, p4, p5*. Waktu eksekusi *Tt* dari simpul *Kt(p1)* adalah *et – st*. Selanjutnya anggaplah proses-proses tersebut dilakukan secara serial, yaitu proses *p1* dimulai pada waktu *t'0* dan berakhir pada waktu *t'1* dan proses *p2* dimulai pada *t'1* dan berakhir pada *t'2* seperti pada Gambar 3.16. Waktu eksekusi ET1 untuk melakukan proses p1 tanpa proses bersamaan lainnya adalah *t'1 – t'0. ET2 = t'2 − t1, ET3 = t'3 – t'2 , ET4 = t'4 – t'3 , dan ET5 = t'5 − t4.* Di sini, waktu eksekusi *Tt = et - st* dari simpul Gambar 3.15 sama dengan ET1 + ET2 + ET3 + ET4 + ET5 dimana proses-proses dilakukan secara serial seperti pada Gambar 3.16. Ini berarti, waktu eksekusi simpul {*pt1, •••, ptm}* adalah sama, yaitu terlepas dari jadwal proses pt1, ••• , ptm dilakukan.



Gambar 3.15. Eksekusi proses yang disisipkan



Gambar 3.16. Eksekusi serial proses

Model Konsumsi Daya

Kami mendapatkan model konsumsi daya dengan mengabstraksi sebagian besar parameter konsumsi daya server *st* dari hasil percobaan yang disajikan pada bagian sebelumnya. Biarkan *CPt(τ)* menjadi kumpulan proses yang sedang dilakukan pada server *st* pada waktu *τ*. Misalkan *CRPt(τ), CWPt(τ),* dan *CCPt(τ) (⊆ CPt(τ))* masing-masing adalah kumpulan proses *R, W,* dan *C* yang terpisah berpasangan, yang secara bersamaan dilakukan pada server st pada waktu τ. Di sini, *CPt(τ) = CRPt(τ) ∪ CWPt(τ) ∪ CCPt(τ).* Misalkan *Nt(τ), NCt(τ), NRt(τ), dan NWt(τ)* adalah banyaknya proses, *|CPt(τ)|, |CCPt(τ)|, |CRPt(τ)|, |CPt(τ)|,* dan *|CWPt(τ)|,* masing-masing.

Tingkat konsumsi daya etA(τ) [W] dari server st untuk melakukan proses pada waktu τ diberikan untuk lingkungan A ∈ {C, W, R, RW, CR, CW, CRW} sebagai berikut:

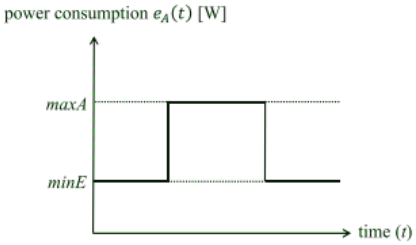
maxRt jika hanya dan setidaknya satu proses R dilakukan di server st pada waktu τ

maxCt jika hanya dan setidaknya satu proses C dilakukan pada server st pada waktu τ

maxCRt jika hanya dan setidaknya satu proses C dan setidaknya satu proses R st dilakukan pada server pada waktu τ

minEt jika tidak ada proses dilakukan pada server st pada waktu τ.

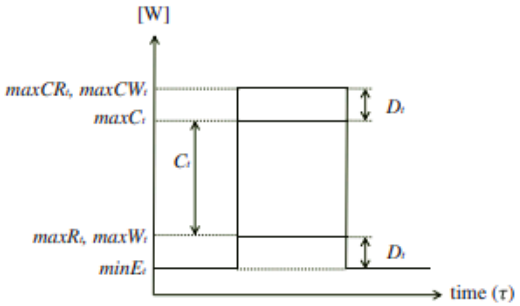
Artinya, tingkat konsumsi daya *etA(τ)* adalah fungsi biner yang mengambil maxAt atau minEt tergantung pada lingkungan A seperti yang ditunjukkan pada Gambar 3.17. Jalan server menggunakan *minEt* tingkat konsumsi daya minimum jika tidak ada proses yang dilakukan pada jalan server. *MinEt* laju konsumsi daya minimum tidak bergantung pada tipe lingkungan A tetapi bergantung pada *st* server.



Gambar 3.17. Tingkat konsumsi daya

Biarkan *Dt* menunjukkan perbedaan tingkat konsumsi daya maksimum *maxRt (= maxWt = maxRWt)* ke *minEt,* yaitu *Dt = maxRt – minEt*. Biarkan *Ct* menjadi *maxCt – minEt*. Laju konsumsi daya maksimum [W] diberikan dalam bentuk *minEt, Dt*, dan *Ct* sebagai berikut [Gambar 3.18]:

- *maksCt = maksEt + Ct*
- *maxRt (= maxWt = maxRWt) = maxEt + Dt.*
- *maxCRt (= maxCWt = maxCWRt) = minEt + Dt + Ct.*



Gambar 3.18. Tingkat konsumsi daya maksimum

Laju konsumsi daya *et(τ)* [W] server st pada waktu τ diberikan sebagai berikut:

$$et(\tau) = minEt + dt(\tau) \cdot Dt + ct(\tau) \cdot Ct \quad (3.12)$$

Di sini, *dt(τ)* dan *ct(τ)* menunjukkan apakah proses R atau W ada atau tidak dan proses C ada atau tidak, masing-masing sebagai berikut:

$$dt(\tau) = \begin{cases} 1, & jika\ NRt(\tau) \geq 1\ atau\ NWt(\tau) \geq 1 \\ 0, & selain\ itu \end{cases}$$
$$Ct(\tau) = \begin{cases} 1, & jika\ NCt(\tau) \geq 1 \\ 0, & selain\ itu \end{cases}$$

Tingkat konsumsi daya *et(τ)* dari server *st* pada waktu τ bergantung pada faktor dt(τ) dan *ct(τ). dt(τ) = 1* jika setidaknya satu proses R atau W dilakukan di server *st* pada waktu τ. Jika tidak ada proses R maupun W yang dilakukan, *dt(τ) = 0. ct(τ) = 1* jika setidaknya satu proses C dilakukan pada server *st* pada waktu τ. Jika tidak, *ct(τ) =* 0.

Berdasarkan hasil percobaan, *minEt* = 100 [W], Dt = 8 [W], dan Ct = 68 [W] pada server st yang spesifikasinya ditunjukkan pada Tabel 3.1.

Konsumsi daya total *Et (st, et)* [Ws] dari server st dari waktu st ke waktu et diberikan sebagai berikut:

$$Et(st, et) = \int_{st}^{et} et(\tau)\, d\tau \quad (3.13)$$

Misalnya, jika hanya proses R yang dilakukan pada server st dari waktu st ke waktu et, konsumsi daya *Et (st, et) = maxRt • (et − st) = (minEt + Dt) • (et− st)* [Ws]. Jika hanya C dan setidaknya satu proses dilakukan pada server *st* dari waktu *st* ke waktu *et, Et (st, et) = maxCt • (et − st) = (minEt + Ct) • (et − st).*

Gambar 3.19 menunjukkan tingkat konsumsi daya *et(τ)* [W] dari server *st* pada waktu τ di mana jenis proses dilakukan secara bersamaan seperti yang ditunjukkan pada Gambar 3.15. Pertama, server *st* mengkonsumsi *minEt* tingkat konsumsi daya minimum karena tidak ada proses yang dilakukan. Kemudian, proses R *p1* dimulai pada waktu *st. et(st) = maxRt* =

minEt + Dt. Selanjutnya, proses *W p2* dimulai pada waktu *st. et(τ) = maksRWt = maksRt (st τ< τ1).* Kemudian, proses *C p3* dimulai pada waktu *t1.*

Di sini, *et(τ) = maxCRWt = minEt + Dt + Ct (τ1 τ< τ2)* karena tiga jenis proses *R, W,* dan *C p1, p2,* dan *p3* masing-masing dilakukan bersamaan. Proses *R p1* berakhir pada waktu *t2. et(τ) = maxCt [W] karena* hanya proses C yang dilakukan ($\tau 2 \leq \tau < \tau 3$). Pada saat *t3*, proses W *p4* dimulai dan kemudian proses *R p6* dimulai. *et(τ) = maksCRWt* ($\tau 3 \leq \tau < \tau 4$). Proses *C p3* berakhir pada waktu *t4. et(τ) = maksRWt* ($\tau 4 \leq \tau < et$). Semua proses berakhir pada waktu et dan konsumsi daya *et(τ) = minEt* ($\tau \geq et$). Area penetasan menunjukkan jumlah total daya listrik yang dikonsumsi pada server *st* pada Gambar 3.19.

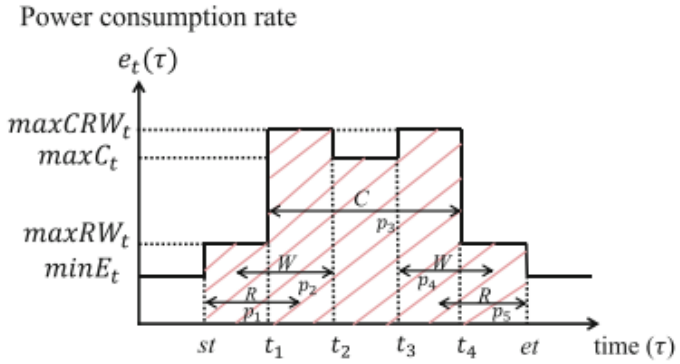
Konsumsi daya total Et (*st, et*) dari server *st* untuk melakukan proses p1, p2, p3, p4, dan p5 adalah:

$$\int_{st}^{et} et(\tau)d\tau =$$

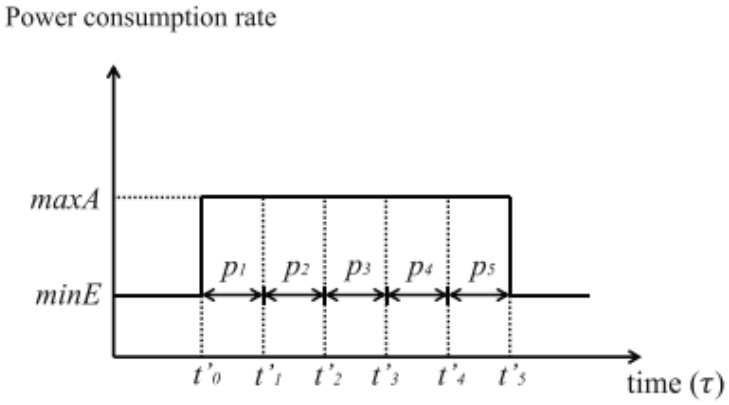
$$\int_{st}^{\tau 1} etRW(\tau)d\tau + \int_{\tau 1}^{\tau 2} etCW(\tau)d\tau + \int_{\tau 2}^{\tau 3} etC(\tau)d\tau + \int_{\tau 3}^{\tau 4} etCW(\tau)d\tau + \int_{\tau 4}^{\tau 5} etRW(\tau)d\tau =$$

$$minEt \cdot (et - st) + Dt \cdot (\tau 2 - st) + Ct \cdot (\tau 3 - \tau 2) + (Dt - Ct) \cdot (\tau 4 - \tau 3) + Dt \cdot (\tau 5 - \tau 4) =$$

$$minEt \cdot (et - st) + Dt \cdot (\tau 2 - st_{\tau 5} - \tau 3) + Ct \cdot (\tau 4 - \tau 1)[WS]$$



Gambar 3.19. Tingkat konsumsi daya



Gambar 3.20. Tingkat konsumsi daya

Gambar 3.20 menunjukkan tingkat konsumsi daya server st untuk jadwal serial Gambar 3.16.

3.6 ALGORITMA PEMILIHAN SERVER

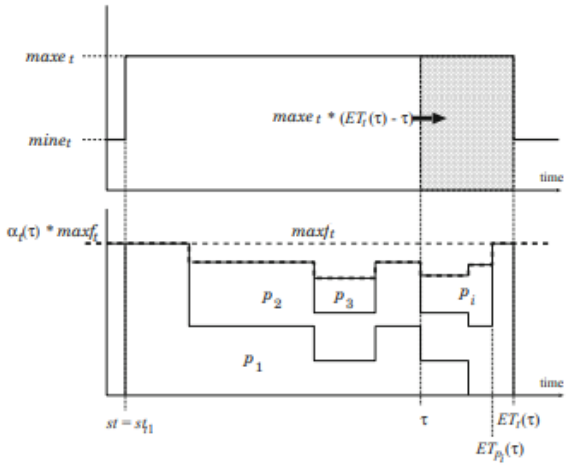
Klien harus mendapatkan layanan yang diperlukan dari satu server dalam satu set S server yang mungkin. Kami membahas algoritma untuk memilih satu server di set server S untuk setiap jenis aplikasi.

Aplikasi CP

Misalkan proses pi dikeluarkan ke penyeimbang beban K pada waktu τ. Pertama, laxity konsumsi daya *(PCL) let(τ)* [Ws] dari setiap st server diperkirakan [Gambar 3.21]:

$$let(\tau) = maxet \cdot (ETt(\tau) - \tau) \qquad (3.14)$$

Dalam algoritma berbasis kelonggaran konsumsi daya (PCLB), server *st* yang mengkonsumsi konsumsi daya minimum dipilih untuk proses *pi,* yaitu *st* server yang *let(τ)* minimum dipilih dalam set server S. Pada waktu *τ,* server *st* dipilih untuk proses pi dengan prosedur berikut PCLB:



Gambar 3.21. Kelemahan konsumsi daya (PCL)

PCLB(τ, S , pi)
{
LEt = ϕ;
for each st **in** S
{
(ETt(τ), ETpi (t)) = **EstimateTermination***(INTt(τ), KPt(τ), pi);*
let(τ) = maxst·(ETt(τ)- τ);
LEi = LEi + {lei(t)};
}
server = st where let(τ) minimum in LEt;
return*(server);*
}

Aplikasi CM

Tarif Transmisi. Pada waktu τ, laju transmisi maksimum *maxtrt(τ)* dari server *st* bergantung pada faktor penurunan transmisi $\sigma t(\tau) = \gamma^{1-|CTt(\tau)|}$ dari *st* server, yaitu jumlah klien yang *st* server secara bersamaan mentransmisikan file pada waktu *τ.* Setiap kali permintaan baru dikeluarkan oleh klien *cs* dan permintaan saat ini untuk klien *cs* diakhiri pada waktu *τ, CTt(τ) = CTt(τ)* ∪ {*cs*} *dan CTt(τ) = CTt(τ) – {cs}* , berturut-turut. Di sini, kecepatan transmisi maksimum *maxtrt(τ)* dari server *st* pada waktu τ dihitung sebagai $\gamma^{1-|CTt(\tau)|}$. *Maxtrt.* Di sini, *0 < γ ≤ 1*. Agar lebih efisien memanfaatkan total laju transmisi *maxtrt(τ),* laju transmisi *trts(τ*) untuk *cv* klien dihitung sebagai berikut:

CalcTR*(st, cv)*
{
V = 0; R = 0; TS = maxtrt(τ) / CTt(τ);
for each client cs **in** CTt(τ)
{
if *Maxrrs* ≤ TS { *trt(τ) = TS and R = R + (TS - Maxrrs);* }
else { *trt(τ) = Maxrrs and V = V + (Maxrrs - TS);* }
}
for each client cs **in** CTt(τ)
{
if *trts(τ) < Maxrrs { trts(τ) = trts(τ) + V • (Maxrrs - trts(τ)) / R; }*
}
}
return*(trtv(τ));*
}

Dalam algoritma **CalcTR(),** bagian yang tidak terpakai dari laju transmisi maksimum server st untuk klien *cs (= trts(τ) - Maxrrs*) dapat digunakan untuk klien lain.

Algoritma Extended Power Consumption-Based (EPCB)

Kami membahas bagaimana penyeimbang beban *K* memilih *st* server untuk klien *cs* di set server *S.* Dalam algoritma berbasis konsumsi daya yang diperluas (EPCB), server *st* dipilih untuk klien *cs* di mana konsumsi daya untuk mengirimkan file ke klien adalah yang terkecil. *ltts(τ)* menunjukkan kelemahan transmisi file *fs* pada *s*t server seperti yang dibahas pada subbagian sebelumnya. Di sini, kelemahan transmisi total server *st* pada waktu *τ* adalah *Σcs ∈CTt (τ)ltts(τ).* Di sini, perkiraan perubahan *EPTts(τ)* [Ws] dari konsumsi daya server *st* untuk mentransmisikan file *f* ke klien *cs* pada waktu τ ketika *st* mulai mentransmisikan file *f* didefinisikan sebagai berikut:

$$EPTts(\tau) = (\sum cs \in CTt(\tau) ltts(\tau)/trts(\tau) \cdot \beta t(|CTt(\tau)| \cdot \delta t \cdot trts(\tau) =$$
$$= \sum cs \in CTt(\tau) ltts(\tau) \cdot \beta t(|CTt(\tau) \cdot \delta t. \quad (3.15)$$

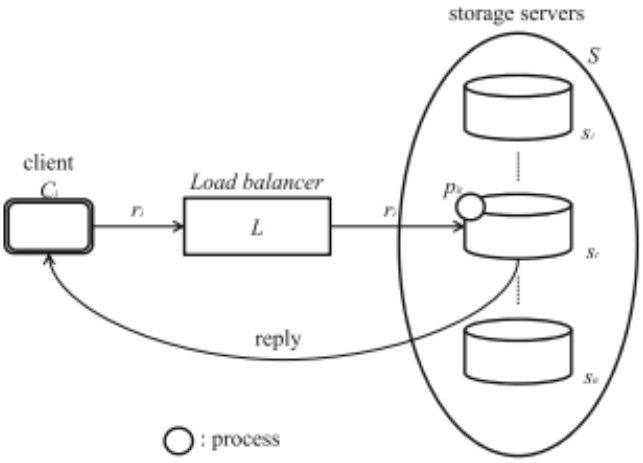
Di sini, server *st* dipilih untuk klien *cs* dalam algoritma EPCB dengan menggunakan *EPTts(τ)* pada waktu *τ* sebagai berikut:

EPCB*(cs, τ) {*
Server = ϕ; $EPT = 0$;
for *each st;* **in** *S {*
EPTts(τ)=Σcs∈ CTt(τ)/tts(τ)· β t(|CTt(τ)|) · δt
if *server = ϕ, {*
server = st;
EPT=EPTts(τ); }
else *{* **if** *EPT > EPTts(τ), {*
EPT=EPTts(τ);
Server=st;
}
}
return*(server);*
}

Misalnya, ada sepasang server *s1* dan *s2*. Koefisien konsumsi daya *δ1* dan *δ2* untuk mengirimkan satu [Mbit] untuk satu klien server *s1* dan *s2* masing-masing adalah 0,09 dan 0,07 [W/Mbit]. Server *s1* dipilih oleh klien *c1 (CT1(τ) = {c1})* dan *s2* dipilih oleh klien *c2 (CT2(τ) = {c2})* masing-masing pada waktu *τ.* Misalkan klien *c3* mengeluarkan permintaan baru untuk mengirimkan file *f* yang ukurannya satu Gbytes ke penyeimbang beban K pada waktu τ. Kelemahan transmisi *lt11(τ)* dan *lt22(τ)* masing-masing adalah 0,1 dan 0,9 [GByte]. Misalkan tingkat peningkatan *β1(2)* dan *β2(2)* dari konsumsi daya server s1 dan s2 masing-masing adalah 1,2 dan 1,1. Di sini, server st yang memiliki nilai minimum dari rumus *Σcs ∈CTt (τ)/tts(τ) • βt(|CTt(τ)|) • δt* dipilih dalam algoritma EPCB. Di sini, *Σcs ∈CT1(τ)/t1s(τ) • β1(|CT1(τ)|) • δ1 = 1,1 • 1,2 0,09 = 0,119. Σcs ∈CT2(τ) lt2s(τ) ·β2(CT2(τ))· δ2 = 1,9 1,1 0,07 = 0,146 [Ws].*
Oleh karena itu, server *s1* dipilih untuk klien *c3* dalam algoritme EPCB.

Aplikasi ST

Kami membahas bagaimana memilih server dalam satu set server penyimpanan untuk setiap permintaan dari klien. Misalkan ada himpunan S dari server *s1, , sn (n ≥ 1).* Klien *ci* pertama-tama mengeluarkan permintaan manipulasi data *ri* ke load balancer *L* seperti yang ditunjukkan pada Gambar 3.22. Load balancer L memilih satu server st di server set *S* dan meneruskan permintaan *ri* ke server *st*. Kemudian, proses pti dibuat untuk permintaan ri dan dijalankan di server *st.* Server *st* mengirimkan balasan dari permintaan *ri* ke klien *ci* setelah menyelesaikan proses *pti.*



Gambar 3.22. Server dan klien

Setiap st server memenuhi model tingkat akses jika jumlah proses yang lebih kecil dilakukan daripada sejumlah *maxNt. St* server kelebihan beban pada waktu τ jika *Nt(τ) > maxNt.* Jika kelebihan beban, model tingkat akses tidak berlaku. Menurut hasil percobaan, *maxNt* adalah 120 untuk st server pada Tabel 3.1. Jika st server kelebihan beban, penyeimbang beban L memilih *su* server lain yang tidak kelebihan beban. Dalam satu algoritma berdasarkan algoritma round robin (RR), load balancer L memilih server st di server set S sebagai berikut:

I. Pertama, t = 1.
II. Jika st server tidak kelebihan beban, pilih st server.
III. Jika st kelebihan beban, t = t + 1 dan lanjutkan ke 2.

Load balancer L meneruskan permintaan ri ke server st. Kemudian, request ri dilakukan sebagai proses pti di server st. Dalam algoritme, sangat penting bagaimana server diurutkan secara total di set server S. Server benar-benar dipesan dalam tingkat konsumsi daya maksimum sebagai m*axCRW1* $\leq$ *maxCRW2* $\leq$ … $\leq$ *maxCRWn*. Artinya, jika server *st* mengkonsumsi daya listrik yang lebih kecil daripada su server lain, *t < u*, yaitu server *st* mendahului *su (st → su).*

Kami menganggap setiap proses adalah salah satu dari jenis proses aplikasi R, W, dan C. Sepasang server pt dan pu di set server S diurutkan dalam dua jenis relasi berikut →RW dan →C:

I. Server st mendahului server lain su sehubungan dengan proses R dan *W (st → RW su)* jika *maxRt < maxRu* atau *ARt* $\geq$ *ARu* jika *maxRt = maxRu,* yaitu *maxWt = maxWu.*
II. Server st mendahului su server lain sehubungan dengan proses *C (st → C su) jika maxCt < maxCu* atau *st* lebih cepat dari *su* jika *maxCt = maxCn.*

Misalnya, server *st* mendahului server lain *su (st → RW su)* jika tingkat konsumsi daya maksimum *maxRt* dari server *st* untuk membaca file lebih kecil dari *maxRu*. Selanjutnya, misalkan *maxRt = maxRu.* Jika tingkat akses baca maksimum *ARt* server *st* lebih besar dari *ARu*,

st → RWsu. Jika tingkat konsumsi daya maksimum *maxCt* lebih kecil dari *maxCu (maxCt < maxCu),* server *st* mendahului server *su (st → C su).* Jika sepasang server *st* dan *su* mengkonsumsi daya yang sama *(maxCt = maxCu), st su* jika *st* lebih cepat dari *su.* Misalkan *S RW* dan *SC* masing-masing adalah sepasang himpunan terurut *S, → RW* dan *S, → C.*

Load balancer L memilih server st di urutan set S RW dan SC sebagai berikut:
I. Awalnya, S R∗W = S RW dan S C∗ = SC.
II. A = RW jika permintaan ri adalah R atau W, A = C jika ri adalah C.
III. Pilih server st sehingga st →A su untuk setiap server lain su di set S ∗A.
IV. Jika st server tidak kelebihan beban, pilih st server dan S A = S A −{st}.
V. Jika tidak (st kelebihan beban), S ∗A = S ∗A −{st}. pergi ke (3).

Jika server st yang dipilih untuk permintaan jenis A pada langkah (2) kelebihan beban, server su sehingga st A su tetapi tiga tidak ada server sv sehingga st A sv A su dipilih di set S ∗A. Kemudian, permintaan ri dari klien ci diteruskan ke server su.

Seperti yang dibahas dalam model konsumsi daya, jika proses C dan R/W dilakukan pada st server, sebagian besar daya maxCRWt dikonsumsi pada st server. Oleh karena itu, jika proses C dilakukan di st server, permintaan R/W tidak dikeluarkan ke st server, S R∗ W = S R∗ W st server lain dipilih di S R∗ W pada langkah (3). Jika tidak, st server diambil. Dengan demikian, permintaan C dan R/W tidak dikirim ke server yang sama.

Set server S R∗ W dan S C∗ adalah kumpulan server yang tidak kelebihan beban dan dipilih sebagai server RW dan C, masing-masing. Jika st server dipilih dalam set server S R∗ W dan S C∗ sesuai dengan algoritme pemilihan dan kelebihan beban, st server dihapus dari set S R∗ W dan S C∗, masing-masing. Untuk setiap kelebihan beban server st, jika proses RW dan C berakhir, st ditambahkan ke set S R∗ W dan S C∗ lagi. S R∗ W = S RW ∪ {S t} dan S C∗ = S C∗ ∪ {S t}, berturut-turut.

3.7 EVALUASI

Kami mengevaluasi algoritma pemilihan server untuk aplikasi CP dan CM dalam hal konsumsi daya total.

Aplikasi CP

Kami mengevaluasi algoritme PCLB dalam hal konsumsi daya total dibandingkan dengan algoritme RR melalui simulasi. Terdapat sepuluh server s1, ..., s10 seperti terlihat pada Tabel 3.5, S = s1, ..., s10. Server di set server S diurutkan sehubungan dengan kecepatan pemrosesan, yaitu tingkat perhitungan maksimum. Di sini, server s1 adalah server tercepat dan server s10 adalah server paling lambat. NCR max ft dari setiap server dipilih secara acak antara 1 dan 0,7 [1/dtk]. Laju perhitungan maksimum max f1 dari server tercepat s1 adalah 1.0 dan max f10 dari server s10 yang paling lambat adalah 0.7. Tingkat degradasi komputasi ɛt dari setiap st server dipilih secara acak antara 0,99 dan 0,95. MinEt tingkat konsumsi daya minimum dari setiap server dipilih secara acak antara 85 dan 150 [W]. MinEt tingkat konsumsi daya maksimum dari setiap server dipilih secara acak antara 1,2 kali dan 1,4 kali dari minEt

tingkat konsumsi daya minimum. Misal NPCR maksimum maxe1 = 1 dan minimum NPCR mine1 = 0,76 untuk server tercepat s1.

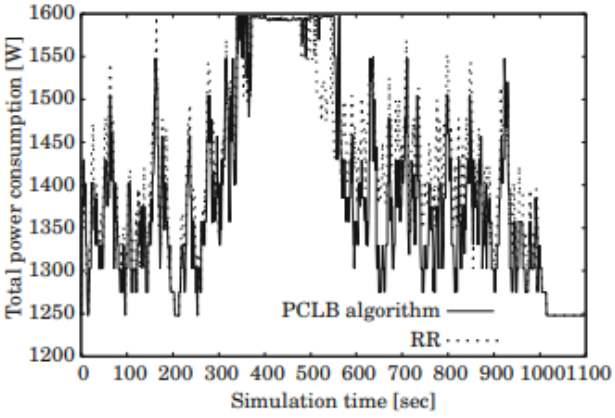
Total 600 jumlah proses, P = p1, ..., p600 dikeluarkan ke server di set server S selama 1.000 detik. MinTi waktu komputasi minimum dari setiap pi proses pada server tercepat dipilih secara acak antara 2 dan 10 [detik] sebagai bilangan real. Misalnya, jika minTi dari proses pi adalah 5,24 [detik], dibutuhkan enam detik untuk server tercepat s1 untuk melakukan proses pi. Di sini, minTti = minTi / max ft untuk setiap server st. Waktu mulai dari setiap proses dipilih secara acak antara 0 dan 1.000 [detik] dalam waktu simulasi. Dalam simulasi, algoritma PCLB dan RR dilakukan pada pola lalu lintas yang sama. Jumlah maksimum proses yang dimulai pada waktu yang sama adalah 5 dan jumlah rata-ratanya adalah 0,6 setiap detik.

Tabel 3.5. Server

Server	*s*1	*s*2	*s*3	*s*4	*s*5
maxft	1.000	0.998	0.859	0.840	0.839
minEt (*minet*)	149 (0.76)	141 (0.72)	133 (0.68)	129 (0.65)	108 (0.55)
maxEt (*maxet*)	197 (1)	186 (0.94)	160 (0.81)	156 (0.79)	136 (0.69)

Server	*s*6	*s*7	*s*8	*s*9	*s*10
maxft	0.833	0.729	0.716	0.713	0.700
minEt (*minet*)	98 (0.5)	139 (0.71)	147 (0.75)	108 (0.55)	96 (0.49)
maxEt (*maxet*)	125 (0.63)	182 (0.92)	196 (0.99)	136 (0.69)	123 (0.62)

Dalam algoritma RR, server diurutkan sebagai s1, ..., s10. Permintaan pertama kali dikeluarkan ke server tercepat s1. Jika server s1 kelebihan beban, permintaan dikeluarkan ke server s2. Dengan demikian, permintaan dikeluarkan ke server di set S.



Gambar 3.23. Konsumsi daya total [Ws]

Gambar 3.23 menunjukkan konsumsi daya total [Ws] dari server di set server S pada setiap waktu τ. Tabel 3.6 menunjukkan total konsumsi daya yang diperoleh dengan mengurangkan konsumsi daya server dalam keadaan diam dari total konsumsi daya selama

simulasi, yaitu konsumsi daya hanya untuk melakukan total 600 proses. Dalam algoritme PCLB, server yang paling sedikit mengkonsumsi konsumsi daya untuk setiap proses pi dipilih. Konsumsi daya total dari algoritma PCLB adalah 167.608 [Ws]. Konsumsi daya total dalam algoritma RR adalah 190.938 [Ws]. Pada algoritma PCLB, konsumsi daya dapat lebih ditekan daripada algoritma RR.

Tabel 3.6. Konsumsi daya total [Ws]

	PCLB	RR
Konsumsi daya total [Ws]	167,608	190,938

Tabel 3.7 menunjukkan total waktu simulasi untuk menghentikan total 600 proses dalam algoritma PCLB dan RR. Waktu terminasi semua proses pada algoritma PCLB dan RR sama yaitu 1.014 [detik]. Perbedaan waktu terminasi antara algoritma PCLB dan RR dapat diabaikan.

Tabel 3.7. Waktu penghentian [dtk]

	PCLB	RR
Waktu penghentian [dtk]	1,014	1,014

Dari hasil evaluasi, kami menyimpulkan konsumsi daya total dapat lebih dikurangi pada algoritma PCLB daripada algoritma RR dan perbedaan waktu terminasi antara algoritma PCLB dan RR dapat diabaikan. Oleh karena itu, algoritma PCLB lebih bermanfaat daripada algoritma RR.

Aplikasi CM

Kami mengevaluasi algoritme EPCB dalam hal konsumsi daya total dan waktu tempuh dibandingkan dengan algoritme RR melalui simulasi. Pada evaluasi terdapat lima buah server S = s1, s2, s3, s4, s5 seperti terlihat pada Tabel 3.8. Koefisien konsumsi daya δt dari setiap server dipilih secara acak antara 0,02 dan 0,11 [W/Mb] berdasarkan hasil percobaan. Laju peningkatan konsumsi daya βt(m) untuk jumlah m klien dipilih secara acak antara 1,09 dan 1,5. MinEt tingkat konsumsi daya minimum dari setiap server dipilih secara acak antara 3 dan 4 [W]. Kecepatan transmisi maksimum Maxtrt dipilih secara acak antara 150 dan 450 [Mbps]. Setiap st server memiliki replika file f dari satu giga-byte.

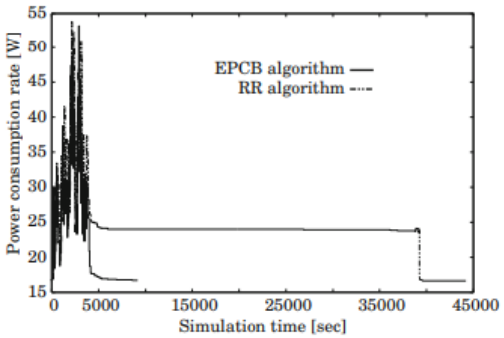
Tabel 3.8. Jenis server

Servers	α	$\beta(m)$	*minE* [W]	*Maxtr* [Mbps]
$s1$	0.03	1.259	3.39	406
$s2$	0.05	1.195	3.17	401
$s3$	0.03	1.285	3.12	249
$s4$	0.09	1.117	3.90	231
$s5$	0.02	1.162	3.02	171

Benar-benar 100 klien mengunduh file f dari satu server st di set server S. Tingkat penerimaan maksimum Maxrrs dari setiap klien cs dipilih secara acak antara 1 dan 100 [Mbps]. Setiap klien cs mengeluarkan permintaan transfer dari file f ke load balancer K pada waktu sts. Di sini, sts waktu mulai dipilih secara acak antara 1 dan 3.600 [detik] pada waktu simulasi. Setiap klien cs mengeluarkan satu permintaan pada waktu sts dalam simulasi algoritma EPCB dan RR.

Gambar 3.24 menunjukkan tingkat konsumsi daya total [W] dari server s1, ..., s5. Tabel 3.9 menunjukkan konsumsi daya total server dalam algoritma EPCB dan RR. Konsumsi daya total algoritma EPCB dan RR masing-masing adalah 204.973 dan 1.058.238 [Ws]. Total konsumsi daya pada algoritma RR lebih besar dari pada algoritma EPCB. Dalam algoritma EPCB, server st dipilih untuk klien cs, yang konsumsi dayanya paling kecil untuk mengirimkan file f ke klien cs.

Tabel 3.10 menunjukkan waktu tempuh pada algoritma EPCB dan RR. Waktu berlalu masing-masing adalah 9.134 [detik] dan 44.178 [detik] dalam algoritme EPCB dan RR. Elapse time algoritma RR lebih lama dibandingkan dengan algoritma EPCB.



Gambar 3.24. Tingkat konsumsi daya total [W]

Tabel 3.9. Konsumsi daya total [Ws]

EPCB	RR
204,973 [Ws]	1,058,238 [Ws]

Tabel 3.10. Waktu berlalu

EPCB	RR
9,134 [sec]	44,178 [sec]

Dari hasil evaluasi, kami menganggap konsumsi daya total dapat lebih ditekan pada algoritma EPCB daripada algoritma RR. Selain itu, elapse time dapat lebih direduksi pada algoritma EPCB daripada algoritma RR. Oleh karena itu, algoritma EPCB lebih bermanfaat daripada algoritma RR.

3.8 KESIMPULAN

Pada bab ini, kita membahas model konsumsi daya server untuk melakukan jenis proses aplikasi berdasarkan hasil percobaan. Pertama, kami mengklasifikasikan aplikasi pada sistem cluster menjadi tiga jenis: perhitungan (CP), komunikasi (CM), dan penyimpanan (ST) jenis aplikasi. Kemudian, kami mengukur konsumsi daya seluruh server untuk menjalankan jenis aplikasi tersebut. Kami mendefinisikan model komputasi, komunikasi, dan konsumsi daya untuk setiap jenis aplikasi berdasarkan pengukuran konsumsi daya server. Model konsumsi daya untuk aplikasi CP, CM, dan ST adalah dua keadaan. Di sini, server mengkonsumsi daya listrik maksimum selama setidaknya satu proses dilakukan. Jika tidak, daya minimum dikonsumsi. Menurut model komputasi, komunikasi, dan konsumsi daya, kami mengusulkan algoritme untuk memilih salah satu server dalam kluster server untuk jenis aplikasi CP dan CM sehingga tidak hanya persyaratan aplikasi yang terpenuhi tetapi juga konsumsi daya total server. dapat dikurangi. Kami mengevaluasi algoritme yang diusulkan untuk aplikasi CP dan CM dalam hal konsumsi daya total dan waktu tempuh dibandingkan dengan algoritme RR. Hasil evaluasi menunjukkan algoritma yang diusulkan dapat mengurangi total konsumsi daya server.

BAB 4
SADAR ENERGI DAN KEAMANAN BERBASIS EVOLUSIONER PENJADWALAN

Memastikan efisiensi energi, keselamatan termal, dan kesadaran keamanan dalam sistem komputasi terdistribusi skala besar saat ini adalah salah satu isu penelitian utama yang mengarah pada peningkatan skalabilitas sistem dan mengharuskan peneliti untuk memanfaatkan pemahaman tentang interaksi antara sistem eksternal. pengguna dan layanan internal dan penyedia sumber daya. Memodelkan interaksi ini dapat menjadi tantangan komputasi terutama dalam infrastruktur dengan akses lokal yang berbeda dan kebijakan manajemen seperti komputasi grid dan cloud. Dalam bab ini, kami mendekati penjadwalan batch independen dalam Computational Grid (CG) sebagai masalah minimalisasi tiga tujuan dengan Makespan, Flowtime, dan konsumsi energi dalam skenario berisiko dan keamanan. Setiap sumber daya fisik dalam sistem dilengkapi dengan modul Dynamic Voltage Scaling (DVS) untuk mengoptimalkan energi daya kumulatif yang digunakan oleh sistem. Efektivitas enam penjadwal grid tunggal dan multi-populasi berbasis genetik telah dibenarkan dalam analisis empiris yang komprehensif.

4.1 PENDAHULUAN

Konsep sistem grid saat ini berkembang jauh melampaui model asli pusat dan jaringan komputasi berkinerja tinggi. Computational Grids (CGs) modern dirancang sebagai infrastruktur kompleks yang terdiri dari ratusan atau ribuan berbagai komponen (komputer, database, dll) dan berbagai layanan dan prosedur yang berorientasi pada pengguna. Pengelolaan infrastruktur semacam itu tetap menjadi masalah yang menantang, terutama ketika personalisasi tambahan dari protokol komunikasi dan keamanan data dan informasi yang ditransfer menjadi isu krusial. Kita dapat mengamati disproporsi yang signifikan dari ketersediaan sumber daya dan penyediaan sumber daya dalam sistem tersebut dan peningkatan konsumsi energi untuk mencocokkan penyediaan sumber daya mengingat tuntutan komputasi. Oleh karena itu, kemampuan penjadwalan aplikasi grid yang efisien dan alokasi sumber daya dalam konfigurasi yang diinginkan dari semua komponen sistem dengan cara yang dapat diskalakan dan kuat sangat penting dalam komputasi grid saat ini.

Penjadwal jaringan cerdas harus secara efisien mengoptimalkan tujuan penjadwalan standar, seperti Makespan, Flowtime, dan pemanfaatan sumber daya, tetapi juga dapat memenuhi persyaratan keamanan pengguna jaringan dan dapat meminimalkan energi yang dikonsumsi oleh semua komponen sistem. Penjadwalan yang hemat energi dan sadar keamanan di CG menjadi upaya yang kompleks karena multi-kendala dan kriteria pengoptimalan yang berbeda serta prioritas yang berbeda dari pemilik sumber daya. Ketegaran komputasi dari masalah panggilan untuk pendekatan heuristik sebagai cara yang efektif untuk merancang penjadwal grid sadar energi tahan risiko dengan pertukaran di antara persyaratan penjadwalan yang berbeda, kendala dan skenario. Literatur terbaru

memanfaatkan kemampuan Genetic Algorithms (GAs) untuk memberikan solusi komputasi hijau yang memuaskan serta penjadwalan yang sadar keamanan dengan mengatasi tantangan penjadwalan yang disebutkan sebelumnya.

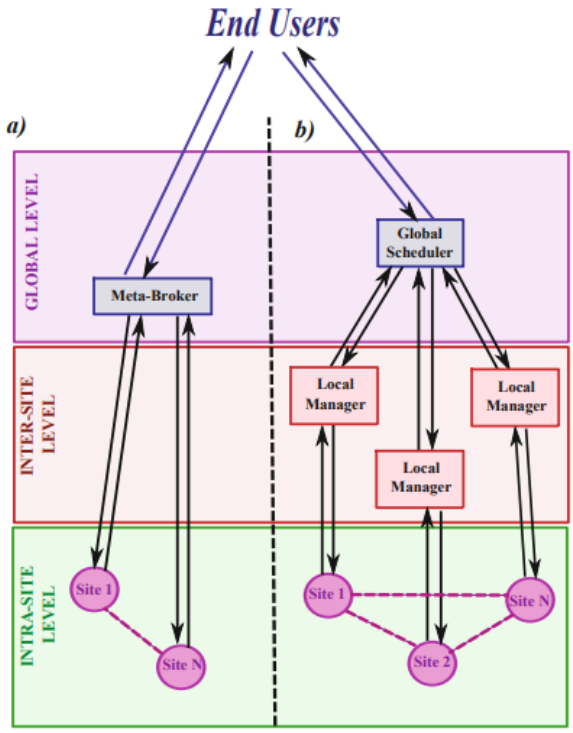
Bab ini membahas masalah Penjadwalan Tugas Batch Independen di CG, di mana tugas diproses dalam mode batch, tidak ada ketergantungan antar tugas, dan dua fungsi tujuan penjadwalan standar, yaitu Makespan dan Flowtime, diminimalkan bersama dengan keamanan dan energi kumulatif konsumsi dalam sistem. Analisis empiris komprehensif dari berbagai metaheuristik berbasis genetik tunggal dan multipopulasi telah disediakan dalam lingkungan grid statis dan dinamis dengan menggunakan simulator grid.

Notasi berikut untuk tugas dan mesin dalam penjadwalan jaringan independen diperkenalkan mulai saat ini dan akan digunakan di seluruh bab ini:

- n – jumlah tugas dalam satu batch;
- m – jumlah mesin yang tersedia dalam sistem untuk pelaksanaan kumpulan tugas tertentu;
- N = {1,..., n} – kumpulan label tugas;
- M = {1,..., m} – himpunan label mesin.

4.2 MODEL GENERIK CLUSTER JARINGAN AMAN

Sistem grid biasanya dimodelkan sebagai arsitektur multi-layer dengan sistem manajemen hierarkis yang terdiri dari dua atau tiga level tergantung pada pengetahuan sistem, akses ke data dan layanan sistem serta sumber daya. Seluruh arsitektur terdiri dari cluster komputasi lokal, seperti yang disajikan pada Gambar. 4.1, dan didefinisikan sebagai kompromi antara sumber daya terpusat dan terdesentralisasi dan manajemen layanan.



Gambar 4.1. Arsitektur cluster dua tingkat (a) dan tiga tingkat (b) dalam Jaringan Komputasi

Arsitektur dua tingkat yang patut dicontoh adalah model Meta-Broker. Dalam model ini Meta-Broker (MB) beroperasi di tingkat global antar-situs dan bertanggung jawab atas pemrosesan semua aplikasi dan informasi yang dikirimkan oleh pengguna jaringan ke administrator sumber daya. Dia juga mengumpulkan informasi umpan balik dari pemilik sumber daya dan mengirimkan hasil perhitungan kepada pengguna.

Model sistem grid tiga tingkat ini terdapat otoritas global pusat (central scheduler) yang bertanggung jawab atas penugasan optimal akhir dari tugas yang diserahkan ke sistem dan komunikasi dengan pengguna jaringan eksternal. Manajer sistem lokal berkomunikasi dengan administrator sumber daya dan pemilik sumber daya dan menentukan "indeks reputasi situs grid" yang selanjutnya diteruskan ke penjadwal global. Semua mesin dan penyedia sumber daya bekerja di tingkat intra-situs lokal.

Struktur hirarkis dari manajemen sistem juga harus digabungkan dengan struktur multi-layer dari komponen grid utama [10], yaitu: (1) lapisan “kain” grid, (2) middleware inti grid, (3) grid lapisan pengguna, dan (4) lapisan aplikasi. Lapisan grid “fabric” terdiri dari sumber daya grid, layanan, dan sistem manajemen sumber daya lokal. Middleware inti grid menyediakan layanan yang berkaitan dengan keamanan dan manajemen akses, penyerahan pekerjaan jarak jauh, penyimpanan, serta informasi dan penjadwalan sumber daya. Lapisan pengguna grid berisi semua pengguna akhir layanan grid dan entitas sistem, dan memainkan peran paling penting dalam memastikan penjadwalan multi-kriteria dan efisiensi manajemen sumber daya.

Peran dari meta-scheduler dan meta-broker di cluster grid berbeda ketika keamanan dianggap sebagai kriteria tambahan dalam proses penjadwalan. Meta-scheduler harus menganalisis persyaratan keamanan yang didefinisikan sebagai parameter permintaan keamanan untuk pelaksanaan tugas dan permintaan pengguna CG untuk sumber daya tepercaya yang tersedia di dalam sistem. Pialang sistem menganalisis indeks “tingkat kepercayaan” dari mesin yang diterima dari manajer sumber daya dan mengirimkan proposal ke penjadwal. Selain itu, broker juga mengontrol alokasi sumber daya dan komunikasi antara pengguna CG dan pemilik sumber daya.

Tingkat kepercayaan dan parameter permintaan keamanan merupakan hasil agregasi dari beberapa penjadwalan dan atribut sistem. Atribut utama dapat ditentukan sebagai berikut:

- Atribut Permintaan Keamanan (Tugas):
  - integrasi data;
  - sensitivitas tugas;
  - kontrol akses;
  - otentikasi rekan;
  - lingkungan pelaksanaan tugas;
- Atribut Tingkat Kepercayaan (Mesin):
  - Atribut Perilaku:
- tingkat keberhasilan pelaksanaan tugas sebelumnya;
- pemanfaatan klaster jaringan kumulatif;
  - Atribut Keamanan Intrinsik:

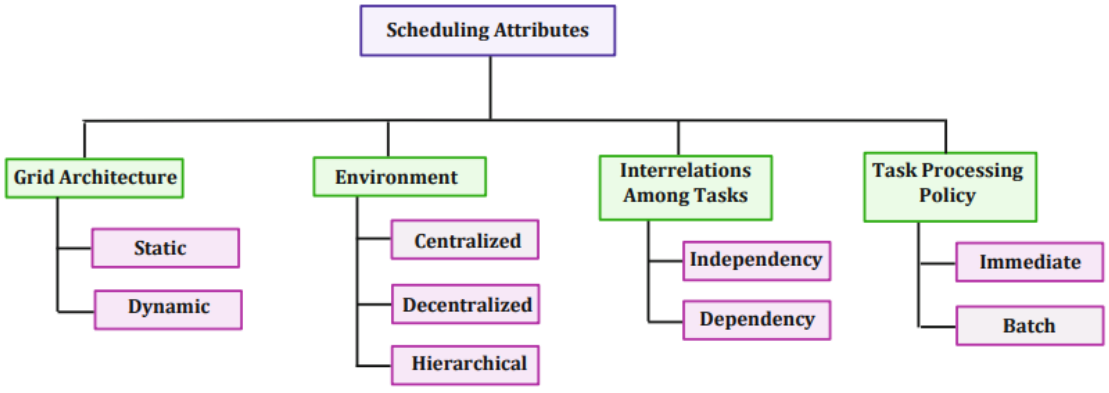
- kemampuan firewall;
- kemampuan deteksi intrusi;
- kemampuan respon intrusi.

Agregasi atribut yang disebutkan di atas menjadi parameter skalar bernilai tunggal dapat direalisasikan dengan menggunakan metode berbasis logika fuzzy seperti yang ditunjukkan oleh Song et al. di [46]. Dalam model ini, mesin dan aplikasi pengguna dicirikan oleh vektor permintaan keamanan S D = [sd1,..., sdn] dan vektor tingkat kepercayaan TL = [tl1,..., tlm]. Nilai parameter sd j dan tli adalah pecahan nyata dalam rentang [0,1] dengan 0 mewakili persyaratan keamanan terendah dan 1 tertinggi untuk eksekusi tugas dan mesin yang paling berisiko dan tepercaya secara berurutan. Sebuah tugas dapat berhasil diselesaikan pada sumber daya ketika kondisi jaminan keamanan terpenuhi. Artinya sdj ≤ tli untuk pasangan tugas-mesin (j, i) yang diberikan. Model Song et al. diterapkan untuk karakteristik mesin dan tugas dalam masalah penjadwalan grid yang didefinisikan pada bagian berikut. Proses pencocokan sd j dengan tli mirip dengan skenario kehidupan nyata di mana pengguna beberapa portal, seperti Yahoo!, diminta untuk menentukan tingkat keamanan sesi login.

4.3 MASALAH PENJADWALAN PADA GRID KOMPUTASI

Ada banyak jenis masalah penjadwalan yang dapat ditentukan dalam lingkungan komputasi yang sangat berparametrik seperti jaringan komputasi. Jenis masalah tertentu dapat didefinisikan sehubungan dengan properti yang berbeda dari lingkungan grid yang mendasarinya dan berbagai kebutuhan pengguna, yaitu (a) jenis lingkungan, (b) jenis arsitektur grid, (c) tugas kebijakan pemrosesan, dan (d) keterkaitan tugas. Untuk mencapai kinerja yang diinginkan dari sistem, semua informasi ini harus “tertanam” ke dalam mekanisme penjadwalan. Keempat atribut penjadwalan tersebut disajikan pada Gambar. 4.2.

Tergantung pada jenis lingkungan grid, penjadwalan dapat direalisasikan sebagai proses statis atau dinamis. Dalam kasus penjadwalan statis, jumlah aplikasi yang diajukan dan sumber daya yang tersedia tetap konstan dalam interval waktu tertentu, sedangkan dalam skenario dinamis, sumber daya dapat ditambahkan atau dihapus dari sistem dengan cara yang tidak dapat diprediksi.



Gambar 4.2. Atribut penjadwalan utama dalam Computational Grids

Manajemen sumber daya dan penjadwalan dapat diatur dalam mode terpusat, terdesentralisasi, atau hierarkis. Dalam model terpusat, ada penjadwal pusat dengan

pengetahuan penuh tentang sistem. Dalam model terdesentralisasi, penjadwal lokal berinteraksi satu sama lain untuk mengelola tugas yang diserahkan ke sistem. Terakhir, dalam model hirarkis, terdapat meta-scheduler sentral, yang berinteraksi dengan manajer lokal (broker) untuk menentukan jadwal yang optimal.

Tugas dalam sistem dapat diproses sesuai dengan kebijakan langsung, di mana tugas dijadwalkan segera setelah dimasukkan ke dalam sistem, atau mode batch, di mana tugas yang dikirimkan dikelompokkan ke dalam batch dan penjadwal menetapkan setiap batch ke sumber daya.

Terakhir, tugas dapat diserahkan dan dihitung secara independen dalam sistem grid atau dianggap sebagai aplikasi paralel dengan batasan prioritas dan keterkaitan antar komponen aplikasi (biasanya dimodelkan dengan Directed Acyclic Graph (DAG)).

Notasi dan Klasifikasi Masalah

Notasi dasar untuk kasus masalah penjadwalan grid dalam CG dapat didefinisikan sebagai berikut:

$$\alpha|\beta|\gamma \tag{4.1}$$

di mana α mencirikan lapisan sumber daya dan tipe arsitektur grid, β menentukan karakteristik pemrosesan dan kendala, dan γ menunjukkan kriteria penjadwalan.

(α) Karakteristik Sumber Daya dan Tipe Arsitektur Grid

Menurut notasi sumber daya standar sumber daya komputasi grid dapat diklasifikasikan sebagai Rm mesin1 dengan kemungkinan kecepatan yang berbeda untuk pekerjaan yang berbeda. tipe arsitektur grid dapat dilambangkan dengan C untuk sentralisasi, D untuk desentralisasi dan H(i) untuk sistem hirarkis, di mana i menunjukkan level sistem. Notasi berikut

$$Pm,H(2) \tag{4.2}$$

digunakan untuk representasi arsitektur grid hierarkis 2 tingkat dengan mesin identik paralel dalam sistem.

(β) Mode Pemrosesan Tugas, Interelasi Tugas, Mode Penjadwalan Statis dan Dinamis, serta Kendala Penjadwalan

Notasi berikut dapat digunakan untuk mengatur atribut tugas grid dan mode penjadwalan (lihat juga Gambar 4.2):

- b – mode kumpulan;
- im – mode langsung;
- dep – ketergantungan antar tugas;
- indep – penjadwalan independen;
- sta – penjadwalan statis;
- dyn – penjadwalan dinamis;

Penjadwalan dapat dilakukan di bawah berbagai kendala yang ditentukan oleh semua pengguna jaringan. Kendala khas termasuk batas anggaran dan tenggat waktu (d j) untuk tugas yang diberikan *j*. Instan penjadwalan batch independen dalam mode dinamis dengan tenggat waktu terbatas dapat dilambangkan sebagai berikut:

b,indep,dyn,dj *(4.3)*

(γ) Kriteria dan Tujuan Penjadwalan

Masalah tugas penjadwalan di CG bersifat multi-tujuan dalam pengaturan umumnya karena kualitas solusi dapat diukur dengan menggunakan beberapa kriteria.

Dua model dasar digunakan dalam optimasi multi-objektif: mode hierarkis dan simultan. Dalam mode simultan (sim) semua tujuan dioptimalkan secara bersamaan sementara dalam kasus hirarkis (hier), tujuan diurutkan secara apriori sesuai dengan kepentingannya dalam model. Proses dimulai dengan mengoptimalkan tujuan yang paling penting dan ketika perbaikan lebih lanjut tidak mungkin dilakukan, tujuan kedua dioptimalkan di bawah batasan menjaga tidak berubah (atau meningkatkan) nilai optimal yang pertama, dan seterusnya. Sangat sulit dalam penjadwalan grid untuk mendefinisikan atau mendekati front Pareto secara efisien, terutama dalam penjadwalan dinamis, di mana ia dapat meluas dengan sangat cepat bersamaan dengan skala grid dan jumlah tugas yang dikirimkan. Kriteria penjadwalan dasar dapat didefinisikan sebagai kriteria pengoptimalan global dan mencakup: Makespan, Flowtime, pemanfaatan sumber daya, penyeimbangan muatan, kedekatan yang cocok, waktu penyelesaian, total waktu penyelesaian berbobot, keterlambatan, jumlah terbobot dari pekerjaan yang terlambat, waktu respons berbobot, dll.

Dalam bab ini kita mempertimbangkan dua kriteria optimasi standar untuk penjadwalan grid, yaitu Makespan dan Flowtime, yang dapat didefinisikan sebagai berikut:

- Makespan – didefinisikan sebagai waktu penyelesaian tugas terbaru dan dapat dihitung dengan rumus berikut:

$$C_{max} = \min_{S \in Schedules} \left\{ \max_{j \in Tasks} C_j \right\}, \qquad (4.4)$$

  di mana Cj menunjukkan waktu ketika tugas j diselesaikan, Tugas menunjukkan kumpulan semua tugas yang diserahkan ke sistem grid dan Schedules adalah kumpulan semua jadwal yang mungkin dihasilkan untuk kumpulan tugas yang dipertimbangkan (atau jumlah tugas saat ini yang diserahkan ke sistem pada saat tertentu); dan

- Flowtime, yang dinyatakan sebagai jumlah waktu penyelesaian semua tugas. Itu dapat didefinisikan dengan cara berikut:

$$C_{sum} = \min_{S \in Schedules} \left\{ \sum_{j \in Tasks} C_j \right\}. \qquad (4.5)$$

  Notasi

$$hier, (C_{max}, C_{sum}) \qquad (4.6)$$

berarti Makespan dan Flowtime dioptimalkan dalam mode hierarkis.

Makespan dan Flowtime menentukan tenggat waktu dan kriteria Kualitas Layanan (QoS) yang biasanya dipertimbangkan di sebagian besar masalah penjadwalan jaringan. Untuk mengekspresikan kesadaran keamanan dan energi, beberapa fungsi tujuan penjadwalan tambahan harus ditentukan. Pada Bagian berikutnya kita mendefinisikan masalah Penjadwalan Batch Independen dan model matriks Waktu yang Diharapkan untuk Menghitung. Tiga skenario penjadwalan utama yang ditentukan dengan mengatur atribut penjadwalan keamanan dan energi dibahas. Tujuan penjadwalan utama dinyatakan dalam waktu penyelesaian mesin.

4.4 MASALAH PENJADWALAN BATCH INDEPENDEN, SKENARIO PENJADWALAN

Penjadwalan Batch Independen adalah salah satu model penjadwalan paling sederhana dan mendasar dalam grid komputasi. Diasumsikan dalam model ini bahwa: (a) tugas dikelompokkan ke dalam batch; (b) tugas dapat dijalankan secara mandiri dalam lingkungan grid statis atau dinamis yang terstruktur secara hierarkis. Menurut notasi yang diperkenalkan di Sec. 4.3.1 (lihat rumus (4.1) contoh dari masalah penjadwalan grid batch independen dapat dinyatakan sebagai berikut:

$$RmH(3)\left[\{b, indep, (stat, dyn), hier\}\right](objectives)), \qquad (4.7)$$

di mana:

- Rm – referensi notasi Graham bahwa tugas dipetakan ke dalam sumber daya (paralel) dari berbagai kecepatan2
- H(3) – menunjukkan sistem manajemen jaringan hirarkis 3 tingkat;
- b – menunjukkan bahwa mode pemrosesan tugas adalah “mode batch”
- indep – menunjukkan “kemandirian” sebagai keterkaitan tugas
- (sta, dyn) – menunjukkan bahwa mode penjadwalan grid statis dan dinamis dipertimbangkan
- hierarki – referensi bahwa tujuan penjadwalan dioptimalkan dalam mode hierarkis
- tujuan – menunjukkan himpunan fungsi tujuan penjadwalan yang dipertimbangkan.

Dalam kasus penjadwalan sadar energi yang disajikan dalam bab ini ada tiga
fungsi tujuan, yaitu:

- Cmax – menunjukkan Makespan yang merupakan tujuan penjadwalan yang dominan;
- Csum – menunjukkan Flowtime sebagai tujuan Quality of Service (QoS);
- EI(EII) – menunjukkan konsumsi energi total (EI atau EII dipilih tergantung pada skenario penjadwalan).

Jadwal direalisasikan dalam mode aman dan berisiko yang ditentukan sesuai dengan persyaratan keamanan yang ditentukan untuk penjadwalan oleh pengguna jaringan atau/dan administrator sistem serta layanan dan sumber daya yang disediakan.

Model Matriks Waktu yang Diharapkan untuk Menghitung (ETC) Diadaptasi untuk Penjadwalan Sadar Energi dan Keamanan dalam Jaringan

Penjadwalan batch independen dalam CG dapat dimodelkan dengan menggunakan model matriks *Expected Time to Compute* (ETC). Dalam versi konvensional model tugas dan mesin ini dicirikan oleh parameter berikut:

(a) Tugas j:

– wlj – parameter beban kerja dinyatakan dalam Jutaan Instruksi (MI)– WL = [wl1,..., wln] adalah vektor beban kerja untuk semua tugas dalam batch;

(b) Mesin i:

– cci – parameter kapasitas komputasi dinyatakan dalam Jutaan Instruksi Per Detik (MIPS), parameter ini adalah koordinat vektor kapasitas komputasi, yang dilambangkan dengan CC = [cc1,..., ccm];

– readyi – waktu siap i, yang menyatakan waktu yang dibutuhkan untuk memuat ulang mesin i setelah menyelesaikan tugas terakhir yang diberikan, vektor waktu siap untuk semua mesin dilambangkan dengan waktu siap = [ready1,..., readym].

Tugas dapat dianggap sebagai aplikasi monolitik atau meta-tugas tanpa ketergantungan antar komponen. Istilah "mesin" terkait dengan unit komputasi tunggal atau multiprosesor atau bahkan ke jaringan area kecil lokal.

Untuk setiap pasangan (j, i) label mesin tugas, koordinat vektor WL dan CC biasanya dihasilkan dengan menggunakan beberapa distribusi probabilitas (kami telah menggunakan distribusi Gaussian untuk tujuan analisis empiris yang disajikan nanti di bab ini) dan digunakan untuk perkiraan waktu penyelesaian tugas j pada mesin i. Waktu penyelesaian ini dilambangkan dengan ETC[j][i] dan dapat dihitung dengan cara berikut:

$$ETC[j][i] = \frac{wlj}{cci} \tag{4.8}$$

Semua parameter ETC[j][i] didefinisikan sebagai elemen matriks ETC, ETC = ETC[j][i] n m, yang merupakan struktur utama dalam model ETC. Elemen dalam baris matriks ETC menentukan perkiraan waktu penyelesaian tugas yang diberikan pada mesin yang berbeda, dan elemen dalam kolom matriks ditafsirkan sebagai perkiraan waktu penyelesaian tugas yang berbeda pada mesin tertentu.

Model ini berguna untuk spesifikasi formal kriteria Makespan dan Flowtime, yang dapat dinyatakan dalam bentuk waktu penyelesaian mesin. Penyelesaian waktu penyelesaian[i] dari mesin i didefinisikan sebagai jumlah parameter waktu siap untuk mesin ini dan waktu eksekusi kumulatif dari semua tugas yang benar-benar ditugaskan ke mesin ini, yaitu:

$$completion[i] = readyi \sum_{j \in Task(i)} ETC[j][i] \tag{4.9}$$

di mana Tugas(i) adalah kumpulan tugas yang diberikan ke mesin i.

Makespan Cmax dinyatakan sebagai waktu penyelesaian maksimal semua mesin, yaitu:

$$C_{max} = \max_{i \in M} completion[i]. \quad (4.10)$$

Waktu alir kumulatif untuk jadwal tertentu didefinisikan sebagai jumlah alur kerja dari urutan tugas pada mesin, yaitu:

$$C_{sum} = \sum_{i \in M} \left[ready_i + \sum_{j \in Sorted[i]} ETC[j][i] \right] \quad (4.11)$$

Di mana S orted[i] menunjukkan serangkaian tugas yang ditetapkan ke mesin i yang diurutkan dalam urutan menaik berdasarkan nilai ETC yang sesuai.

Waktu penyelesaian mesin dan waktu pelaksanaan tugas yang berhasil pada mesin ini dapat berubah jika kondisi keamanan dan pengoptimalan energi juga dipertimbangkan. Oleh karena itu, model matriks ETC konvensional harus diperluas dengan memasukkan karakteristik tambahan dari tugas dan mesin ke dalam sistem, dan modifikasi metodologi pembuatan matriks ETC.

Kondisi Keamanan

Dalam penjadwalan sadar-keamanan, semua "aktor" sistem, yaitu pengguna akhir grid, penjadwal grid, layanan cluster dan penyedia sumber daya, administrator sistem, dan pemilik sumber daya dapat menentukan persyaratan mereka sendiri untuk pelaksanaan tugas yang aman pada mesin grid dan akses aman ke data dan layanan grid. Persyaratan pengguna yang paling penting dapat dinyatakan dengan cara berikut:

- Akses ke Data Jarak Jauh: Data input dan output yang ditentukan oleh pengguna dapat disimpan dari jarak jauh. Oleh karena itu, pengguna perlu memberikan lokasi data jarak jauh. Jika sistem file area luas di mana-mana beroperasi di grid, pengguna hanya perlu memperhatikan lokasi file dan data sehubungan dengan beberapa lokasi root di mana mereka disimpan.
- Spesifikasi Sumber Daya: Pengguna dapat menentukan kebutuhan individualnya untuk sumber daya yang diperlukan dalam mengoptimalkan waktu pelaksanaan dan biaya tugas penjadwalan dan komputasi. Pengguna mungkin ingin menargetkan jenis sumber daya tertentu (misalnya mesin SMP), tetapi tidak harus peduli dengan jenis manajemen sumber daya di jaringan, atau dengan sistem manajemen sumber daya pada sumber daya individu di jaringan.
- Keandalan sumber daya: Dalam beberapa kasus, mesin dalam sistem grid mungkin tidak tersedia karena dinamika sistem yang tinggi atau kebijakan khusus dari pemilik sumber daya. Pengguna harus diberi tahu tentang keandalan sumber daya untuk mengurangi biaya kemungkinan kegagalan sumber daya atau penghentian tugas yang dijalankan. Jika terjadi kegagalan sumber daya, administrator sistem dapat

mengaktifkan prosedur penjadwalan ulang atau migrasi tugas, dan kebijakan pencegahan.

- Keandalan Sumber Daya: Pengguna mungkin diminta untuk mengalokasikan tugasnya di sumber daya yang paling tepercaya. Oleh karena itu, pengguna harus dapat memverifikasi indeks kepercayaan sumber daya dan memperkirakan permintaan keamanan untuk tugasnya pada sumber daya yang tersedia.
- Persyaratan mekanisme autentikasi dan otorisasi standar: Pengguna kemungkinan besar akan menggunakan skema autentikasi sertifikat standar. Sertifikat dapat ditandatangani secara digital oleh otoritas sertifikat, dan disimpan di repositori pengguna, yang dikenali oleh sumber daya dan pemilik sumber daya. Sertifikat diinginkan dibuat secara otomatis oleh aplikasi antarmuka pengguna selama pengiriman tugas.

Keberhasilan pelaksanaan tugas yang diserahkan ke CG mungkin tidak mungkin (atau terputus) jika persyaratan tersebut sangat kuat dan akses ke sumber daya terbatas. Di sisi lain, cluster grid atau sumber daya grid mungkin tidak dapat diakses oleh meta-scheduler global atau pengguna grid ketika terinfeksi dengan intrusi atau serangan berbahaya. Ini berarti bahwa beberapa, bahkan sangat sederhana, protokol otorisasi dan otentikasi dan beberapa mekanisme perlindungan anti-virus diperlukan untuk penjadwalan yang efisien terutama di lingkungan yang dinamis. Dalam kasus seperti itu, mesin dan tugas juga harus ditandai dengan tingkat kepercayaan tli dan keamanan menuntut parameter sd j.

Mari kita nyatakan dengan Prf(a) Matriks Probabilitas Kegagalan Mesin, yang mendefinisikan probabilitas kegagalan mesin selama eksekusi tugas Pf[j][i] untuk setiap pasangan tugas-mesin (j, i). Probabilitas ini dapat dihitung dengan menggunakan fungsi distribusi eksponensial negatif, yaitu:

$$Prf[j][i] = \begin{cases} 0, & sdj \leq tli \\ 1 - e^{-\alpha(sdj - tli)}, & sdj > tli \end{cases} \quad (4.12)$$

di mana α ditafsirkan sebagai koefisien kegagalan dan merupakan parameter global model. Penjadwal dapat menginisialisasi pekerjaannya dalam dua mode berbeda: (a) mode aman, di mana ia menganalisis elemen matriks Probabilitas Kegagalan Mesin untuk meminimalkan probabilitas kegagalan untuk pasangan tugas-mesin; dan (b) mode berisiko, di mana dia melakukan penjadwalan "biasa" tanpa analisis awal dari kondisi keamanan, membatalkan penjadwalan tugas jika terjadi kegagalan mesin, dan menjadwal ulang tugas ini di sumber lain.

Modus Aman

Dalam skenario ini, semua kondisi keamanan dan keandalan sumber daya diverifikasi untuk semua pasangan tugas-mesin (j, i). Tujuan utama dari meta-scheduler adalah untuk merancang jadwal yang optimal, di luar kriteria lainnya, kemungkinan kegagalan mesin selama eksekusi tugas akan minimal. Diasumsikan bahwa "biaya" tambahan dari verifikasi kondisi jaminan keamanan untuk pasangan mesin-tugas tertentu dapat menunda perkiraan waktu pelaksanaan tugas pada mesin. Biaya ini kira-kira sebanding dengan kemungkinan kegagalan

mesin selama pelaksanaan tugas. Waktu penyelesaian mesin i dalam mode aman dilambangkan dengan penyelesaian[i] dan dapat dihitung sebagai berikut:

$$completion[i] = readyi + \sum_{\{j \in Tasks(i)\}} (1 + Prf[j][i]ETC[j][i]) \quad (4.13)$$

Makespan dan Flowtime dalam hal ini dapat dinyatakan sebagai berikut:

$$CsMax = maxcompletions[i] \quad (4.14)$$

$$CsSum = \sum_{i \in M} \left[readyi + \sum_{j \in Sorted\,[i]} (1 + Prf[j][i] \cdot ETC[j][i] \right] \quad (4.15)$$

Modus Berisiko

Dalam skenario ini semua kondisi aman dan gagal diabaikan. Proses penjadwalan diwujudkan sebagai prosedur dua langkah. Pertama, penjadwalan dilakukan hanya dengan menganalisis matriks ETC konvensional. Jika kegagalan mesin diamati, maka tugas yang belum selesai untuk sementara dipindahkan ke set backlog tugas. Tugas dari rangkaian ini dijadwalkan ulang sesuai aturan yang ditentukan untuk mode aman. Total waktu penyelesaian mesin i (i ∈ M) dalam hal ini dapat didefinisikan sebagai berikut:

$$completion^r[i] = completion[i] + completion^s_{res}[i], \quad (4.16)$$

di mana penyelesaian[i] dihitung dengan menggunakan Persamaan. (4.9), untuk tugas-tugas yang terutama ditugaskan ke mesin i , dan penyelesaian [i] adalah waktu penyelesaian mesin i yang dihitung dengan menggunakan Persamaan. (4.13) untuk tugas yang dijadwalkan ulang, yaitu tugas yang ditetapkan ulang ke mesin i dari sumber daya lainnya. Makespan dan Flowtime dalam mode ini dapat dihitung dengan cara berikut:

$$C^r_{max} = \max_{i \in M} completion^r[i]. \quad (4.17)$$

$$C^r_{sum} = \sum_{i \in M} \left[ready_i + \sum_{j \in Sorted[i]} ETC[j][i] + \sum_{j \in Sorted_{res}[i]} \left(1 + Pr_f[j][i]\right) \cdot ETC[j][i] \right]. \quad (4.18)$$

Dalam mode optimisasi hirarkis (lihat Persamaan (4.7), hierarki parameter) di mana Makespan didefinisikan sebagai kriteria penjadwalan yang dominan, fungsi tujuan Flowtime harus diminimalkan dalam skenario yang aman dan berisiko yang tunduk pada kendala berikut:

- dalam mode aman

$$ready_i + \sum_{j \in Sorted[i]} \left(1 + Pr_f[j][i]\right) \cdot ETC[j][i] \leq C^s_{max}) \; \forall i \in M; \quad (4.19)$$

- dalam mode berisiko

$$ready_i + \sum_{j \in Sorted[i]} ETC[j][i] + \sum_{j \in Sorted_{res}[i]} \left(1 + Pr_f[j][i]\right) \cdot ETC[j][i] \leq C^r_{max} \forall i \in M. (4.20)$$

Meskipun kemungkinan kegagalan mesin diharapkan lebih tinggi dalam mode berisiko daripada dalam mode aman, tentu saja tidak ada jaminan keberhasilan pelaksanaan semua tugas dalam skenario keamanan. Dapat diamati bahwa jika kondisi jaminan keamanan terpenuhi untuk setiap pasangan tugas-mesin (yaitu *sd j* ≤ *tli* untuk *i* ∈ *M, j* ∈ *N*), waktu penyelesaian mesin dalam mode aman dan berisiko identik dengan waktu penyelesaian didefinisikan untuk masalah penjadwalan independen standar, di mana diasumsikan bahwa setiap tugas harus berhasil dijalankan pada setiap mesin dan tidak ada persyaratan keamanan yang dianalisis.

Model Energi

Model energi yang disajikan dalam bab ini didasarkan pada teknik *Dynamic Voltage and Frequency Scaling* (DVFS) yang digunakan untuk menyesuaikan suplai tegangan dan frekuensi node komputasi grid.

Setiap mesin di grid dilengkapi dengan modul DVFS untuk menskalakan voltase suplai dan frekuensi operasinya. Diasumsikan bahwa frekuensi mesin sebanding dengan kecepatan pemrosesannya. Ini mengikuti dari Persamaan. (4.24) bahwa pengurangan tegangan suplai dan frekuensi berkorelasi langsung dengan pengurangan penggunaan energi. Tabel 4.1 menunjukkan parameter tipikal untuk 16 level DVFS dan tiga kategori "energik" utama untuk mesin.

Tabel 4.1. Level DVFS untuk tiga kelas mesin

Class 1			Class 2		Class 3	
Level	**Volt**	**Rel Freq**	**Volt**	**Rel.Freq**	**Volt**	**Rel.Freq**
0	1.5	1.0	2.2	1.0	1.75	1.0
1	1.4	0.9	1.9	0.85	1.4	0.8
2	1.3	0.8	1.6	0.65	1.2	0.6
3	1.2	0.7	1.3	0.50	1.9	0.4
4	1.1	0.6	1.0	0.35		
5	1.0	0.5				
6	0.9	0.4				

Kelas energik mesin i, (i ∈ M) dilambangkan dengan si dan diwakili oleh vektor Vr(i) dari level DVFS, yang dapat ditentukan sebagai berikut:

$$Vr_{(i)} = \left[(v_{s_0}(i), f_{s_0}(i)); \ldots; (v_{s_{l(max)}}(i), f_{s_{l(max)}}(i))\right]^T \quad (4.21)$$

di mana vsl (i) mengacu pada suplai tegangan untuk mesin *i* pada level *sl, fsl (i)* adalah parameter penskalaan untuk frekuensi mesin pada level yang sama sl, dan lmax adalah jumlah

level di kelas *si.* Parameter *fs0 (i),..., fsl(max) (i)* terletak pada kisaran [0,1] dan harus ditafsirkan sebagai frekuensi relatif mesin i dari kelas si pada s0,..., sl(maks) level DVFS.

Pengurangan frekuensi mesin dan tegangan suplainya dapat menyebabkan perpanjangan waktu komputasi dari tugas yang dijalankan pada mesin tersebut. Untuk pasangan "mesin-tugas" (j, i), waktu penyelesaian tugas j pada mesin i pada level DVFS yang berbeda di kelas si dapat diinterpretasikan sebagai koordinat vector $\widehat{ETC}[j][i]$ yang didefinisikan dengan cara berikut:

$$\widehat{ETC}[j][i] = \left[\frac{1}{f_{s_0}(i)} \cdot ETC[j][i], \ldots, \frac{1}{f_{s_{l(max)}}(i)} \cdot ETC[j][i]\right], \qquad (4.22)$$

di mana ETC[j][i] adalah waktu penyelesaian yang diharapkan untuk tugas j pada mesin i yang dihitung dengan menggunakan model matriks ETC konvensional.

Dalam penjadwalan hemat energi, waktu penyelesaian yang dihitung untuk setiap pasangan (j, i) dari label tugas-mesin dalam matriks ETC konvensional (lihat Persamaan (4.8)) harus diganti dengan vektor $\widehat{ETC}[j][i]$, yang adalah untuk mengatakan:

$$\widehat{ETC} = \left[\widehat{ETC}[j][i][s_l]\right]_{n \times m \times s_{l(max)}}, \qquad (4.23)$$

dimana EˆT C[j][k][sl] adalah waktu yang diperlukan untuk menyelesaikan tugas j pada mesin i pada level sl.

Model DVFS didasarkan pada model konsumsi daya yang digunakan dalam rangkaian logika semikonduktor oksida logam komplementer (CMOS). Dalam model CMOS, daya kapasitif Pow ji yang digunakan oleh mesin i untuk menghitung tugas j dinyatakan sebagai berikut:

$$Pow_{ji} = A \cdot C \cdot v^2 \cdot f, \qquad (4.24)$$

di mana A adalah jumlah sakelar per siklus clock, C adalah beban kapasitansi total, v adalah tegangan suplai dan f adalah frekuensi mesin. Energi yang dikonsumsi per mesin i untuk perhitungan tugas j didefinisikan sebagai hasil integrasi berikut:

$$E_{ji} = \int_0^{completion[j][i]} Pow_{ji}(t)dt, \qquad (4.25)$$

di mana penyelesaian[j][i] adalah waktu penyelesaian tugas j pada mesin i. Berdasarkan Persamaan (4.23) energi yang digunakan untuk menyelesaikan tugas j pada mesin i pada level sl dapat didefinisikan sebagai berikut:

$$E_{ji}(s_l) = \gamma \cdot (f_{s_l}(i))_j \cdot f \cdot [(v_{s_l}(i))_j]^2 \cdot \widehat{ETC}[j][i][s_l], \qquad (4.26)$$

di mana γ = A C adalah parameter konstan untuk kelas mesin tertentu, (vsl (i)) j adalah nilai suplai tegangan untuk kelas si dan mesin i pada level sl untuk tugas komputasi j, dan (fsl (i)) j adalah a frekuensi relatif yang sesuai untuk mesin i.

Oleh karena itu waktu komputasi untuk setiap kemungkinan pasangan (j,i) pada level sl dapat dihitung sebagai berikut:

$$Tim_{\{j,i,s_l\}} = \gamma \cdot (f_{s_l}(i))_j \cdot f \cdot [(v_{s_l}(i))_j]^2 \cdot (f_{s_1}(i))_j \cdot ETC[j][i] = \\ = \gamma \cdot f \cdot [(v_{s_l}(i))_j]^2 \cdot ETC[j][i] \quad (4.27)$$

Energi kumulatif yang digunakan oleh mesin i untuk menyelesaikan semua tugas dari batch yang ditugaskan ke mesin ini, ditentukan sebagai berikut:

$$E_i = \sum_{\substack{j \in Tasks(i) \\ l \in \hat{L}_j}} \{Tim_{\{j,i,s_l\}}\} + \gamma \cdot f \cdot [v_{s_{max}}]^2 \cdot ready_i + \gamma \cdot f_{s_{min}}(i) \cdot f \cdot \\ \cdot [v_{s_{min}}(i)]^2 \cdot Idle[i] = \gamma \cdot f \cdot \sum_{\substack{j \in Tasks(i) \\ l \in \hat{L}_i}} ([(v_{s_l}(i))_j]^2 \cdot ETC[j][i]) + \quad (4.28) \\ + [v_{s_{max}}(i)]^2 \cdot ready_i + f_{s_{min}}(i) \cdot [v_{s_{min}}(i)]^2 \cdot Idle[i]$$

di mana Idle[i] menunjukkan waktu idle mesin i, dan L^i menunjukkan subset dari level DVFS yang digunakan untuk tugas yang ditetapkan ke mesin i3.

Akhirnya, energi kumulatif rata-rata yang digunakan oleh sistem grid untuk penyelesaian semua tugas dalam batch didefinisikan sebagai:

$$E_{batch} = \frac{\sum_{i=1}^{m} E_i}{m} \quad (4.29)$$

Model ini digunakan pada bagian berikut untuk spesifikasi dua skenario penjadwalan "energik".

Skenario Penjadwalan Sadar Energi

Diasumsikan mesin yang disertakan dengan modul DVFS dapat bekerja dalam dua mode berikut:

I. Mode Max-Min, di mana setiap mesin bekerja pada tingkat DVFS maksimal selama eksekusi dan komputasi tugas dan beralih ke mode siaga setelah eksekusi semua tugas yang diberikan ke mesin ini;

II. Mode Catu Daya Modular, di mana setiap mesin dapat bekerja pada level DVFS yang berbeda selama eksekusi tugas dan kemudian dapat masuk ke mode siaga.

Prosedur perhitungan dan optimalisasi Makespan, Flowtime, dan energi kumulatif yang digunakan oleh sistem, berbeda dalam skenario penjadwalan yang disebutkan di atas dan juga dalam mode aman dan berisiko.

Dalam Mode Max-Min, rumus untuk waktu penyelesaian, Makespan, dan Flowtime sama dengan Persamaan. (4.16), (4.17) dan (4.18) dalam mode berisiko; dan Persamaan. (4.13), (4.14) dan (4.15) dalam mode aman.

Waktu idle untuk mesin i yang bekerja dalam Mode Max-Min dapat dinyatakan sebagai perbedaan antara Makespan dan waktu penyelesaian mesin ini, yaitu:

$$Idle^r[i] = C^r_{max} - completion^r[i] \quad (4.30)$$

dalam mode berisiko, dan

$$Idle^s[i] = C^s_{max} - completion^s[i] \quad (4.31)$$

dalam mode aman.

Dalam kasus mesin dengan waktu penyelesaian maksimal (Makespan), faktor menganggur adalah nol.

Dalam Mode Catu Daya Modular, untuk setiap pasangan mesin tugas, level DSV sl harus ditentukan. Rumus untuk menghitung waktu penyelesaian dalam skenario aman dan berisiko pada level si dapat didefinisikan sebagai berikut:

$$completion^s_{II}[i] = ready_i + \sum_{j \in Tasks(i)} \frac{1}{f_{s_I}(i)} \cdot \left(1 + P_f[j][i]\right) \cdot ETC[j][i]. \quad (4.32)$$

dalam mode aman, dan

$$completion^r_{II}[i] = completion_{II}[i] + completion^s_{res,II}[i] \quad (4.33)$$

dalam mode berisiko, di mana penyelesaian [i] adalah waktu penyelesaian yang dihitung untuk tugas yang dijadwalkan ulang pada mesin i, dan penyelesaian II[i] dihitung sebagai berikut:

$$completion_{II}[i] = ready_i + \sum_{j \in Tasks(i)} \frac{1}{f_{s_I}(i)} \cdot ETC[j][i]. \quad (4.34)$$

Rumus ini digunakan untuk spesifikasi Makespan (Crmax)II dan (Csmax)II baik dalam mode keamanan maupun mode berisiko. Flowtime dapat dihitung dari Persamaan yang dimodifikasi. (4.18) dan (4.15) dengan mengganti komponen *ECT[j][i]* dengan $\frac{1}{fsj(i)} \bullet ETC[j][i].$ Terakhir, rumus waktu idle dapat dinyatakan sebagai berikut:

$$Idle^s_{II}[i] = \left(C^s_{max}\right)_{II} - completion^s_{II}[i] \quad (4.35)$$

$$Idle^r_{II}[i] = \left(C^r_{max}\right)_{II} - completion^r_{II}[i] \quad (4.36)$$

Energi rata-rata yang dikonsumsi dalam sistem dalam **Mode Min-Max** dan skenario berisiko ditentukan sebagai berikut:

$$E^r_I = \frac{1}{m} \cdot \sum_{i=1}^{m} \gamma \cdot completion^r[i] \cdot f \cdot [v_{s_{max}}(i)]^2 + \\ + \frac{1}{m} \cdot \sum_{i=1}^{m} \gamma \cdot f_{s_{min}}(i) \cdot [v_{s_{min}}(i)]^2 \cdot Idle^r[i] \quad (4.37)$$

Dalam Mode Catu Daya Modular, energi kumulatif rata-rata diberikan oleh Persamaan. (4.29):

$$E_{II} = \frac{\sum_{i=1}^{m} E_i}{m} \quad (4.38)$$

di mana Ei dihitung dengan memodifikasi Persamaan. (4.28) dengan menggunakan Persamaan. (4.33)–(4.36). Prosedur pengoptimalan adalah dua langkah. Pertama, Makespan dan Flowtime diminimalkan. Kemudian, menjaga dua tujuan pertama pada tingkat optimal, energi kumulatif yang digunakan dalam sistem juga diminimalkan.

4.5 PENJADWAL BATCH BERBASIS GENETIK YANG SADAR KEAMANAN

Metode heuristik terkenal dari ketangguhannya dan telah berhasil diterapkan untuk menyelesaikan masalah penjadwalan dan masalah optimisasi kombinatorial umum di berbagai bidang. Dalam penjadwalan jaringan, metode ini dapat menangani berbagai atribut penjadwalan dan aspek energi dan keamanan tambahan.

Metode penjadwalan heuristik biasanya diklasifikasikan menjadi tiga kelompok utama, yaitu (1) berbasis kalkulus (algoritma rakus dan metode ad-hoc); (2) stokastik (metode terbimbing dan tidak terarah); dan (3) metode pencacahan (pemrograman dinamis dan algoritma cabang-dan-terikat). Dalam karya ini kami fokus pada metode berbasis genetik yang diklasifikasikan sebagai penjadwal stokastik berbasis populasi. Algoritma 1 mendefinisikan kerangka umum untuk algoritma genetika (GA) yang dirancang untuk penjadwalan grid. Kerangka kerja ini didasarkan pada model umum GA populasi tunggal konvensional yang didedikasikan.

Algoritma 1. Template mesin genetik untuk HGS-Sched
1: Hasilkan P0 populasi awal dengan ukuran μ; e = 0
2: Evaluasi P0;
3: sementara tidak terminasi-kondisi lakukan
4: Pilih kumpulan parental Te dengan ukuran λ; T t := S pilih(Pe);
5: Lakukan prosedur crossover pada par individu di Te dengan probabilitas pc; Pe := Silang(Te);
6: Lakukan prosedur mutasi pada individu di Pec dengan probabilitas pm; Pe := c m Mutasi(Pe); Pm
7: Evaluasi Pm
8: Buat populasi baru Pe+1 dengan ukuran μ dari individu di Pe dan Pe ; Pe+1 := M Ganti (Pe; Pe)
9: e := e + 1;
10: akhir sementara
11: kembali Individu terbaik yang ditemukan sebagai solusi;

Parameter e dalam Alg. 1 adalah penghitung generasi di GA (e menghitung jumlah loop di GA). Populasi dalam algoritma ini dikodekan dengan menggunakan dua metodologi berikut:

• Pengkodean Langsung: Vektor jadwal S dinyatakan dalam bentuk:

$$S = [i_1, \ldots, i_n]^T, \quad (4.39)$$

di mana i j M menunjukkan nomor mesin tempat tugas yang diberi label j dijalankan.

• Pengodean Berbasis Permutasi: Vektor jadwal didefinisikan sebagai berikut:

$$Sch = [Sch_1, \ldots, Sch_n]^T, \quad (4.40)$$

dimana S chi ∈ N, i = 1,..., n . Jadwal S ch adalah vektor label tugas yang ditugaskan ke mesin. Untuk setiap mesin, label tugas yang diberikan ke mesin ini diurutkan dalam urutan menaik menurut waktu penyelesaian tugas.

Dalam representasi berbasis permutasi, beberapa informasi tambahan tentang jumlah tugas yang diberikan ke setiap mesin diperlukan. Jumlah total tugas yang ditugaskan ke mesin i dilambangkan dengan S chi dan ditafsirkan sebagai koordinat ke-i T dari vektor penugasan S˜ch = [S˜ch1,..., S˜chm] , yang mendefinisikan sebenarnya banyak mesin grid. Representasi langsung digunakan untuk jadwal pengkodean pada populasi dasar Pe dan Pe+1, dan representasi permutasi digunakan pada populasi Pe dan Pt cm.

Populasi awal di Algoritma 1 dihasilkan dengan menggunakan Waktu Penyelesaian Minimum + Pekerjaan Terpanjang hingga Sumber Daya Tercepat - Pekerjaan Terpendek menjadi Sumber Daya Tercepat Metode MTC + LJFR-SJFR, di mana semua kecuali dua individu dihasilkan secara acak. Kedua individu tersebut diciptakan dengan menggunakan Longest Job to Fastest Resource - Shortest Job to Fastest Resource (LJFR-SJFR) dan heuristik Minimum Completion Time (MCT). Dalam metode LJFR-SJFR awalnya jumlah m tugas dengan beban kerja tertinggi ditugaskan ke m mesin yang tersedia diurutkan dalam urutan menaik dengan kriteria kapasitas komputasi. Kemudian sisa tugas yang belum ditetapkan dialokasikan ke mesin tercepat yang tersedia. Dalam heuristik MCT, tugas yang diberikan ditugaskan ke mesin yang menghasilkan waktu penyelesaian paling awal.

Algoritma 1 disesuaikan dengan masalah penjadwalan CG melalui penerapan metode pengkodean khusus dan operator genetik. Operator dari set berikut digunakan dalam percobaan yang disajikan dalam buku ini:

- Operator pemilihan: Pemeringkatan Linear;
- Operator Crossover: Crossover yang Dipetakan Sebagian (PMX) dan Cycle Crossover (CX);
- Operator mutasi: Pindahkan, Tukar dan Rebalancing;
- Operator pengganti: *Steady State, Elitist Generational, Struggle*.

Semua operator yang disebutkan di atas biasanya digunakan dalam meta-heuristik genetik yang didedikasikan untuk memecahkan masalah optimisasi kombinatorial.

Dalam metode Linear Ranking probabilitas seleksi untuk setiap individu dalam suatu populasi sebanding dengan peringkat individu tersebut. Peringkat individu terburuk didefinisikan sebagai nol, sedangkan peringkat terbaik didefinisikan sebagai pop size 1, dimana pop size adalah ukuran populasi.

Operator crossover dan mutasi berikut diimplementasikan untuk representasi jadwal berbasis permutasi. Dalam Partially Matched Crossover (PMX) segmen dari satu kromosom induk dipetakan ke segmen kromosom induk lainnya (posisi yang sesuai) dan gen yang tersisa ditukar sesuai dengan 'hubungan' pemetaan tugas dan mesin yang ditentukan. oleh aturan

penjadwalan beton. Tn Cycle Crossover (CX), pertama, siklus alel diidentifikasi. Operator crossover membiarkan siklus tidak berubah, sedangkan segmen yang tersisa di string parental dipertukarkan.

Dalam Pindahkan mutasi, tugas dipindahkan dari satu mesin ke mesin lainnya. Meskipun tugas dapat dipilih dengan tepat, strategi mutasi ini cenderung membuat jumlah pekerjaan per mesin tidak seimbang. Ini diwujudkan dengan modifikasi dua koordinat pada vektor u˜ dari kode jadwal dalam representasi berbasis permutasi. Ide utama dari metode Rebalancing adalah untuk meningkatkan solusi (dengan menyeimbangkan ulang beban mesin) dan kemudian memutasikannya. Dalam prosedur penyeimbangan ulang, pertama, mesin yang paling kelebihan beban dipilih. Dua tugas j dan ˆj diidentifikasi dengan cara berikut: j ditugaskan ke mesin lain i∗, ˆj ditugaskan ke i dan ETC[j][i∗] ETC[ˆj][i]. Kemudian penugasan dipertukarkan untuk tugas j dan ˆj. Dalam mutasi Swap, indeks dari dua tugas yang dipilih dalam representasi jadwal ditukar.

Populasi dasar untuk loop GA baru di Algoritma 1 dapat didefinisikan dengan menggunakan metode penggantian generasi Elit, di mana populasi baru berisi dua solusi terbaik dari populasi basis lama dan sisanya adalah keturunan yang baru dihasilkan. Dalam metode penggantian Steady State, himpunan keturunan terbaik (jumlah elemen dalam himpunan ini tetap) menggantikan solusi terburuk dalam populasi dasar lama. Kelemahan utama dari metode ini adalah dapat menyebabkan konvergensi prematur dari algoritma dalam beberapa solusi lokal. Mekanisme penggantian Struggle dapat menjadi alat yang efektif untuk menghindari konvergensi penjadwal yang terlalu cepat ke optima lokal. Dalam metode tersebut, generasi baru individu diciptakan dengan mengganti sebagian populasi dengan individu yang paling mirip – jika penggantian ini meminimalkan nilai fitness. Definisi prosedur penggantian perjuangan memerlukan spesifikasi ukuran kesamaan yang sesuai, yang menunjukkan tingkat kesamaan antara dua kromosom GA. Kami menggunakan dalam pekerjaan ini jarak Mahalanobis untuk mengukur jarak antara jadwal menurut rumus berikut:

$$sim_e(S^1;S^2) = \sqrt{\sum_{j=1}^{n} \frac{(S^1[j] - S[j])^2}{\sigma_P^2}} \qquad (4.41)$$

di mana σP adalah standar deviasi dari S 1[j] terhadap populasi P.

Strategi perjuangan telah terbukti sangat efektif dalam memecahkan beberapa masalah multi-tujuan skala besar. Namun, biaya komputasi bisa sangat tinggi, karena kebutuhan untuk menghitung jarak antara semua mata air di populasi yang dihasilkan dan individu dalam populasi dasar untuk loop GA saat ini. Untuk mengurangi waktu eksekusi dari prosedur perjuangan kami menggunakan teknik hash, di mana tabel hash dengan kunci alokasi sumber daya tugas dibuat. Nilai kunci ini dihitung sebagai jumlah nilai absolut dari pengurangan setiap posisi dan presedennya dalam representasi langsung dari vektor jadwal (membaca vektor jadwal secara melingkar).

Delapan belas varian GA dengan semua kemungkinan kombinasi dari operator-operator ini dijelaskan dalam Tabel 4.2.

Tabel 4.2. Delapan belas varian penjadwal berbasis GA populasi tunggal

Penjadwal	Metode silang	Metode mutasi	Metode penggantian
GA-PMX-M-SS	Dicocokkan Sebagian (PMX)	Pindah	Stabil
GA-PMX-M-EG	Dicocokkan Sebagian (PMX)	Pindah	Generasi Elit
GA-PMX-M-ST	Dicocokkan Sebagian (PMX)	Pindah	Berjuang
GA-PMX-S-SS	Dicocokkan Sebagian (PMX)	Menukar	Stabil
GA-PMX-S-EG	Dicocokkan Sebagian (PMX)	Menukar	Generasi Elit
GA-PMX-S-ST	Dicocokkan Sebagian (PMX)	Menukar	Berjuang
GA-PMX-R-SS	Dicocokkan Sebagian (PMX)	Menyeimbangkan kembali	Stabil
GA-PMX-R-EG	Dicocokkan Sebagian (PMX)	Menyeimbangkan kembali	Generasi Elit
GA-PMX-R-ST	Dicocokkan Sebagian (PMX)	Menyeimbangkan kembali	Berjuang
GA-CX-M-SS	Siklus (CX)	Pindah	Stabil
GA-CX-M-EG	Siklus (CX)	Pindah	Generasi Elit
GA-CX-M-ST	Siklus (CX)	Pindah	Berjuang
GA-CX-S-SS	Siklus (CX)	Menukar	Stabil
GA-CX-S-EG	Siklus (CX)	Menukar	Generasi Elit
GA-CX-S-ST	Siklus (CX)	Menukar	Berjuang
GA-CX-R-SS	Siklus (CX)	Menyeimbangkan kembali	Stabil
GA-CX-R-EG	Siklus (CX)	Menyeimbangkan kembali	Generasi Elit
GA-CX-R-ST	Siklus (CX)	Menyeimbangkan kembali	Berjuang

Semua algoritma ini telah dievaluasi secara empiris dengan menggunakan simulator grid. Hasil analisis empiris ini disajikan pada bagian berikutnya.

4.6 EVALUASI EMPIRIS PENJADWAL JARINGAN GENETIK

Pada bagian ini kami menyajikan analisis empiris sederhana dari efisiensi 18 implementasi penjadwal GA populasi tunggal yang didefinisikan pada bagian sebelumnya dalam meminimalkan fungsi tujuan penjadwalan yang ditentukan dalam bagian 4.4.

Performa relatif dari semua penjadwal telah dihitung dengan empat metrik berikut:

- Makespan – kriteria penjadwalan dominan yang dapat dihitung dengan berbagai cara tergantung pada kriteria keamanan dan energik;
- Mean Flowtime – rata-rata Flowtime dihitung sebagai berikut:

$$Mean\_Flowtime = \frac{C_{sum}}{m} \tag{4.42}$$

dimana Csum mirip dengan Makespan, dapat dihitung dengan berbagai cara tergantung pada kriteria keamanan dan energik;

- tingkat peningkatan konsumsi energi relatif dinyatakan sebagai berikut:

$$Im(E) = \frac{E_I - E_{II}}{E_{batch}} \cdot 100\%, \tag{4.43}$$

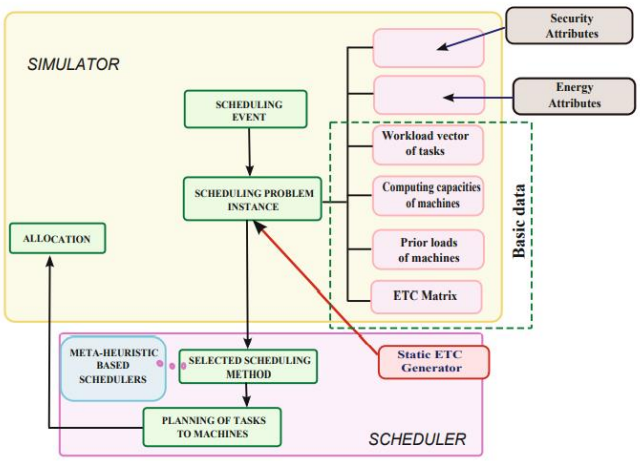
di mana EII dan EI didefinisikan dalam Persamaan. (4.29) dan Persamaan. (4.37) berturut-turut;

- Parameter FailureRate Failr didefinisikan sebagai berikut:

$$Fail_r = \frac{n_{failed}}{n} \cdot 100\% \tag{4.44}$$

dimana nfailed adalah jumlah tugas yang belum selesai, yang harus dijadwal ulang.

Untuk menyediakan eksperimen dan mensimulasikan lingkungan grid kami menggunakan simulator grid Sim-G-Batch, yang didasarkan pada model berbasis peristiwa diskrit HyperSim-G, dan memfasilitasi evaluasi heuristik penjadwalan yang berbeda un - der berbagai kriteria penjadwalan di beberapa skenario grid. Skenario ini ditentukan oleh konfigurasi kondisi keamanan untuk penjadwalan dan akses ke sumber jaringan, ukuran jaringan, parameter pemanfaatan energi, dan dinamika sistem. Simulator memungkinkan aktivasi atau penonaktifan yang fleksibel dari semua kriteria dan modul penjadwalan, serta bekerja dengan campuran penjadwal meta-heuristik. Konsep utama dan alur umum versi hemat energi dan keamanan dari simulator Sim-G-Batch disajikan pada Gambar 4.3.



Gambar 4.3. Diagram Alir Umum Simulator Sim-G-Batch yang Sadar Keamanan dan Energi Terkait dengan Penjadwalan

Tabel 4.3. Nilai parameter kunci dari simulator grid dalam kasus statis dan dinamis

	Medium	Very Large
Static case		
Nb. of hosts	64	256
Resource cap.	$N(5000, 875)$	
Total nb. of tasks	1024	4096
Workload of tasks	$N(250000000, 43750000)$	
Dynamic case		
Init. hosts	64	256
Max. hosts	70	264
Min. hosts	50	240
Resource cap.	$N(5000, 875)$	
Add host	$N(562500, 84375)$	$N(437500, 65625)$
Delete host	$N(625000, 93750)$	
Init. tasks	768	3072
Total tasks	1024	4096
Workload	$N(250000000, 43750000)$	
Both cases		
Security demands sd_j	$U[0.6; 0.9]$	
Trust levels tl_i	$U[0.3; 1]$	
Failure coefficient α	3	

Set dasar dari input data untuk simulator meliputi:

- vektor beban kerja tugas,
- vektor kapasitas komputasi mesin,
- vektor beban mesin sebelumnya, dan
- matriks ETC dari estimasi waktu eksekusi tugas pada mesin.

Set ini diperluas oleh vektor SD dan Tl untuk spesifikasi parameter keamanan utama dan atribut "energi" berikut:

- parameter spesifikasi kategori mesin (jumlah kelas, nilai kapasitas komputasi maksimal, interval rentang kapasitas komputasi untuk setiap kelas, parameter kecepatan operasional mesin untuk setiap kelas, dll.);
- Matriks level DVFS untuk kategori mesin.

Tabel 4.4. Setelan GA untuk tolok ukur statis dan dinamis yang besar

Parameter	Elitist Generational/Struggle	Steady State
degree of branches (t)	0	
period_of_metaepoch (α)	$1/2 * n/10$	$(2 * n/10$
nb_of_metaepochs	10	
population size (pop_size)	$\lceil (\log_2 n)^2 - \log_2 n \rceil$	$4 * (\log_2 n - 1)$
intermediate pop.	$pop\_size - 2$	$(pop\_size)/3$
cross probab.	0.8	1.0
mutation probab.	0.2	
max_time_to_spend	40 sec. (*static*) / 75 sec. (*dynamic*)	

Simulator ini sangat parametrized untuk mencerminkan banyak skenario grid yang realistis. Semua penjadwal dipisahkan dari badan utama simulator dan diimplementasikan sebagai perpustakaan eksternal. Performa dari semua metaheuristik berbasis GA telah dianalisis dalam dua skenario ukuran grid: Sedang dengan 64 host dan 1024 tugas, dan Sangat Besar dengan 256 host dan 4096 tugas. Kapasitas sumber daya dan beban kerja tugas dihasilkan secara acak oleh distribusi Gaussian. Nilai sampel dari parameter input kunci yang digunakan dalam eksperimen untuk simulasi disajikan pada Tabel 4.3.

Mesin dapat bekerja pada 16 level DVFS dan dapat dikategorikan ke dalam tiga kelas sumber daya "energik", Kelas I, Kelas II, dan Kelas III. Pengidentifikasi kelas telah dipilih secara acak untuk mesin. Nilai tegangan suplai dan frekuensi mesin relatif pada semua level DVFS ditentukan dalam Tabel 4.1. Parameter kunci untuk penjadwalan genetik disajikan pada Tabel 4.4.

Hasil

Setiap percobaan diulang 30 kali di bawah konfigurasi operator dan parameter yang sama. Hasil untuk Makespan, Flowtime, tingkat kegagalan, dan tingkat peningkatan konsumsi energi relatif yang dirata-ratakan dalam 30 kali menjalankan simulator disajikan pada Tabel 4.5–4.11.

Dari hasil tersebut dapat disimpulkan bahwa kualitas penjadwalan sangat bergantung pada kombinasi operasi crossover dan mutasi yang tepat. Dalam semua kasus yang dipertimbangkan, crossover PMX bersama dengan mutasi Move memberikan hasil terburuk dan kombinasi crossover CX dengan mutasi Rebalancing tampaknya menjadi yang paling efektif di sebagian besar kasus. Memang, dalam kasus statis, algoritme GA CX R ST menempati urutan pertama pada sekitar 75% instans dan algoritme GA CX R SS adalah yang terbaik dalam 25% sisanya dari total instans. Dalam skenario dinamis, situasinya serupa: algoritme GA CX R ST mencapai hasil terbaik di sekitar 62,5% instans dan GA CX R SS – di 27,5% sisanya. Algoritme GA CX R EG dalam semua kasus mendapat peringkat sebagai penjadwal terbaik kedua atau ketiga. Saya dapat mengamati bahwa dalam kelompok algoritme dengan operator mutasi dan persilangan yang sama, mekanisme penggantian Struggle memiliki dampak positif terbaik pada kinerja algoritme. Tingkat kegagalan rata-rata untuk algoritma yang dipertimbangkan berada di kisaran 37% – 54%, dan tingkat peningkatan energi berada di kisaran 8% – 36%.

Tabel 4.5. Rata-rata nilai Makespan untuk delapan belas penjadwal berbasis GA dalam skenario aman [±s.d.], (s.d. = standar deviasi)

Strategy	Medium	Very Large	Medium	Very Large
GA-PMX-M-SS	4165583.135 [± 690668.957]	4288022.184 [± 824291.171]	4294782.139 [± 911521.151]	4466888.391 [± 560336.505]
GA-PMX-M-EG	4177553.793 [± 190066.130]	4301234.468 [± 451450.111]	4279562.342 [± 375819.231]	4485562.553 [± 750553.639]
GA-PMX-M-ST	4157425.782 [± 379356.512]	4295435.433 [± 515425.756]	4278653.544 [± 489524.252]	4435334.722 [± 364469.977]
GA-PMX-S-SS	4126691.373	4268733.837	4202645.677	4418525.678
	[± 7592355.678]	[± 575235.535]	[± 5733466.268]	[± 953456.457]

GA-PMX-S-EG	4148405.275 [± 43761.015]	4285671.797 [± 605962.237]	4216301.090 [± 53486.153]	4421427.663 [± 701992.116]
GA-PMX-S-ST	4133822.183 [± 380836.642]	4261593.241 [± 634705.248]	4199261.431 [± 555304.633]	439189.653 [± 711735.974]
GA-PMX-R-SS	4099722.422 [± 939509.617]	4209834.534 [± 505720.705]	4122903.467 [± 149437.882]	4332736.028 [± 563164.075]
GA-PMX-R-EG	4111018.744 [± 837452.949]	4217551.633 [± 540610.017]	4161359.893 [± 590881.527]	4375643.075 [± 748754.806]
GA-PMX-R-ST	4096915.832 [± 653833.933]	4202740.834 [± 468878.756]	4109927.347 [± 997874.784]	4313319.834 [± 752973.821]
GA-CX-M-SS	4057635.783 [± 711365.726]	4175408.673 [± 796525.362]	4034538.827 [± 933600.267]	4284402.632 [± 684131.232]
GA-CX-M-EG	4067320.534 [± 654607.977]	4184339.256 [± 582147.023]	4090546.734 [± 751364.567]	4303769.235 [± 952170.387]
GA-CX-M-ST	404176222.362 [± 987536.453]	4167744.674 [± 803122.735]	4009105.874 [± 679674.456]	4241630.356 [± 728082.735]
GA-CX-S-SS	3954447.634 [± 886725.393]	4125317.764 [± 671196.560]	39872916.546 [± 998157.753]	4205709.634 [± 564696.828]
GA-CX-S-EG	4004965.356 [± 918998.520]	4156258.356 [± 713229.061]	3990734.764 [± 594713.746]	4222631.465 [± 957740.279]
GA-CX-S-ST	3940265.563 [± 989958.718]	4098563.182 [± 455636.448]	41614537.863 [± 56051.579]	4219725.327 [± 988429.144]
GA-CX-R-SS	392372.215 [± 384195.783]	405397.326 [± 697330.373]	3985763.378 [± 581510.863]	4152623.367 [± 712805.735]
GA-CX-R-EG	3912183.536 [± 341357.564]	4095293.987 [± 981563.633]	4093277.356 [± 754356.222]	4197426.546 [± 774433.987]
GA-CX-R-ST	**3907233.453 [± 552722.245]**	**3982865.453 [± 827733.653]**	**3900433.453 [± 368863.563]**	**4100641.722 [± 704973.453]**

Tabel 4.6. Nilai Makespan rata-rata untuk delapan belas penjadwal berbasis GA dalam skenario berisiko [±s.d.], (s.d. = standar deviasi)

Strategy	Medium	Very Large	Medium	Very Large
GA-PMX-M-SS	4183064.456 [± 303594.681]	41996544.866 [± 633541.835]	4272342.367 [± 523861.891]	4351579.387 [± 955889.372]
GA-PMX-M-EG	4196660.637 [± 795391.867]	4207559.346 [± 673312.285]	4250769.729 [± 570063.523]	4378943.912 [± 663276.562]
GA-PMX-M-ST	41666682.922 [± 557522.757]	4182062.259 [± 659041.716]	4242506.997 [± 564267.992]	4342520.961 [± 435640.377]
GA-PMX-S-SS	4150255.659 [± 364204.730]	4156125.373 [± 612692.293]	4239582.165 [± 566626.175]	4323745.969 [± 685834.282]
GA-PMX-S-EG	4149105.038 [± 440578.157]	416379.853 [± 472235.032]	4220166.253 [± 837511.681]	4309346.443 [± 515978.598]
GA-PMX-S-ST	4133090.502 [± 271571.051]	4177107.429 [± 473557.935]	4198308.287 [± 492853.075]	4294855.678 [± 430583.789]
GA-PMX-R-SS	4126676.780 [± 728966.192]	4118808.678 [± 649630.565]	4164756.208 [± 56766.709]	4254796.820 [± 580607.466]

GA-PMX-R-EG	4129602.691 [± 270662.077]	4109778.902 [± 913428.430]	4182790.083 [± 761523.938]	4270973.780 [± 100219.228]
GA-PMX-R-ST	4063022.741 [± 430715.830]	4077928.331 [± 410172.944]	4158207.631 [± 611305.330]	4232768.084 [± 499217.899]
GA-CX-M-SS	4096232.073 [± 554669.949]	4093218.110 [± 602133.299]	4114422.323 [± 721311.202]	4188744.263 [± 672746.995]
GA-CX-M-EG	4109712.181 [± 869717.384]	4099788.995 [± 453095.636]	4141965.814 [± 817008.729]	4209289.242 [± 636853.293]
GA-CX-M-ST	4054429.908 [± 464589.755]	4070157.151 [± 446428.587]	4127500.870 [± 870899.646]	4198327.481 [± 631520.079]
GA-CX-S-SS	4022572.196 [± 516330.313]	4032408.549 [± 590815.709]	4090392.967 [± 898536.995]	4129281.695 [± 917216.254]
GA-CX-S-EG	4015442.255 [± 405905.662]	4033181.392 [± 650601.916]	4106452.250 [± 982489.800]	4164853.468 [± 785931.696]
GA-CX-S-ST	4006934.189 [± 436482.946]	4010678.953 [± 516992.488]	4039201.986 [± 816514.067]	4140525.379 [± 633513.047]
GA-CX-R-SS	3986872.198 [± 806632.171]	3972483.354 [± 618035.814]	4024651.986 [± 436431.848]	4109893.365 [± 747400.818]
GA-CX-R-EG	3999111.928 [± 692024.001]	4002987.835 [± 467779.063]	4089790.735 [± 632965.292]	4091857.736 [± 791598.934]
GA-CX-R-ST	**3955725.386 [± 911937.937]**	**39677356.846 [± 879753.092]**	4030546.634 [± 850836.928]	**4038634.546 [± 932835.453]**

Tabel 4.7. Nilai Flowtime rata-rata untuk delapan belas penjadwal berbasis GA dalam skenario aman [±s.d.], (s.d. = standar deviasi)

Strategy	Medium	Very Large	Medium	Very Large
GA-PMX-M-SS	2193567211.836 [± 128370293.469]	8309597733.826 [± 291748525.988]	2399334723.236 [± 411586354.582]	8401592320.927 [± 889213844.016]
GA-PMX-M-EG	2206734272.356 [± 135504556.429]	8326428289.314 [± 13744410.910]	2404322051.418 [± 439654339.019]	8410389312.587 [± 989122999.052]
GA-PMX-M-ST	2182232499.779 [± 218224825.514]	8297351808.190 [± 207359074.643]	2357752153.775 [± 476389087.741]	8395922739.597 [± 606291276.896]
GA-PMX-S-SS	2158140985.025 [± 180302378.918]	8261813008.415 [± 240332696.510]	2316860418.277 [± 593778909.154]	8367766411.152 [± 755794904.240]
GA-PMX-S-EG	2175542598.454 [± 162987312.683]	828568131.930 [± 369229026.877]	2332618330.048 [± 45367872.253]	8388110055.663 [± 507151151.344]
GA-PMX-S-ST	2147247760.724 [± 211590465.457]	8271025623.997 [± 453137510.918]	2320515589.851 [± 711100452.764]	8373206007.045 [± 784801239.205]
GA-PMX-R-SS	21142225326.145 [± 240067553.885]	8249643747.642 [± 253050662.960]	2286055243.299 [± 577523722.350]	8338732539.636 [± 741000998.450]
GA-PMX-R-EG	2136071183.723 [± 143892082.090]	8258614783.371 [± 233840747.951]	2292713481.576 [± 377833946.994]	835216627.023 [± 699126484.932]
GA-PMX-R-ST	2117878614.800 [± 196255094.628]	8227564670.521 [± 225294411.337]	2283126157.824 [± 543390178.534]	8322161405.019 [± 736033399.656]
GA-CX-M-SS	2105676446.564 [± 18596758.737]	8195359732.256 [± 282655389.940]	2263563557.141 [± 539894530.830]	8294397268.110 [± 790327708.149]
GA-CX-M-EG	2125655077.793	8238431289.893	2274715011.673	8306547838.652

	[± 142293067.762]	[± 212224990.911]	[± 604531703.308]	[± 812904982.726]
GA-CX-M-ST	21111848829.461 [± 110335889.154]	8204221008.299 [± 254605827.121]	2238467253.243 [± 513415557.850]	8272449832.295 [± 822664629.425]
GA-CX-S-SS	2084338418.886 [± 165598016.948]	8179625719.400 [± 250682728.242]	2204403080.180 [± 430849430.300]	8242964769.102 [± 600614818.712]
GA-CX-S-EG	2097868914.479 [± 17614361.064]	8190763974.690 [± 334857074.927]	2227164271.644 [± 473657175.059]	8260315502.581 [± 569231789.391]
GA-CX-S-ST	2078132035.030 [± 120473769.993]	8166781141.473 [± 250660461.860]	2195675260.221 [± 577482551.646]	8253290991.790 [± 769767684.495]
GA-CX-R-SS	2060575340.426 [± 123314394.677]	8138208217.698 [± 772885985.554]	**2177096342.984 [± 489373983.094]**	**8211987409.256 [± 999265736.534]**
GA-CX-R-EG	2066071183.009 [± 103578848.015]	8143852396.768 [± 272146853.672]	2196929745.169 [± 533701714.987]	8235408309.560 [± 795752579.317]
GA-CX-R-ST	**2035128734.875 [± 55234984.937]**	**8109087253.984 [± 339791854.937]**	2199758911.356 [± 436450229.234]	8221050176.985 [± 708270871.387]

Tabel 4.8. Nilai Flowtime rata-rata untuk delapan belas penjadwal berbasis GA dalam skenario berisiko [±s.d.], (s.d. = standar deviasi)

Strategy	Medium	Very Large	Medium	Very Large
GA-PMX-M-SS	2343724728.245 [± 598538835.360]	8321846250.347 [± 431346593.814]	2413245472.632 [± 417986755.794]	8644534678.245 [± 404482983.284]
GA-PMX-M-EG	2389239424.349 [± 129097532.581]	8347268532.324 [± 432955978.981]	2436061767.548 [± 701148099.631]	8660435684.376 [± 664054585.165]
GA-PMX-M-ST	2307355142.051 [± 126906586.696]	8308727439.878 [± 256156869.233]	2403766189.879 [± 469026059.421]	8605175251.450 [± 693219859.321]
GA-PMX-S-SS	2263649999.867 [± 268317752.960]	8262608448.916 [± 216787358.248]	2360160544.425 [± 656197729.295]	8502620979.464 [± 851502055.485]
GA-PMX-S-EG	2286784630.430 [± 184783757.268]	8292058041.397 [± 499461667.383]	2393131389.346 [± 454805534.425]	8531596508.841 [± 573812983.895]
GA-PMX-S-ST	2237247760.724 [± 244213856.135]	8271025623.997 [± 434633012.674]	2377515589.851 [± 788991696.341]	8513206007.045 [± 719513808.715]
GA-PMX-R-SS	2226429310.580 [± 255806730.308]	8229357685.290 [± 290693183.985]	2333974209.902 [± 519849125.536]	8472673073.563 [± 743945636.397]
GA-PMX-R-EG	2213613356.567 [± 167571940.967]	8254728732.731 [± 283645546.676]	2348829866.347 [± 526061100.837]	8493687750.198 [± 784339554.900]
GA-PMX-R-ST	2151930817.794 [± 134663685.381]	8184860096.988 [± 242728273.618]	2320407124.381 [± 528711756.114]	8466099314.428 [± 728490457.382]
GA-CX-M-SS	2188284638.934 [± 162112136.850]	8199577483.835 [± 257276315.811]	2239455642.778 [± 387064110.517]	8426829568.357 [± 537095310.247]
GA-CX-M-EG	2193593276.050 [± 163978670.974]	8211445673.176 [± 173050142.794]	2296573568.612 [± 621630992.871]	8457381627.647 [± 596961998.050]
GA-CX-M-ST	2132978230.586 [± 129547599.824]	8166440897.331 [± 182827503.146]	2270646610.327 [± 616490962.213]	8432573052.527 [± 885691236.077]
GA-CX-S-SS	2126523158.454 [± 228154433.283]	8161517451.690 [± 277432894.907]	2222289600.939 [± 552469744.108]	8365526025.239 [± 773215992.243]
GA-CX-S-EG	2113772162.066 [± 187159613.747]	8154839647.617 [± 326230384.931]	2233669538.605 [± 620551577.626]	8392873916.020 [± 859689519.119]
GA-CX-S-ST	2106340445.440	8146356979.846	2218990182.691	8345528446.458

	[± 138247946.733]	[± 239873730.726]	[± 525735729.121]	[± 947608913.414]
GA-CX-R-SS	2076534164.177 [± 181982147.103]	**81127123653.774** **[± 19799961.957]**	221654378.495 [± 48988254.993]	**8291082340.932** **[± 834522826.826]**
GA-CX-R-EG	2098943746.287 [± 143403467.472]	8138965387.563 [± 176627923.813]	2208567534.205 [± 564619337.921]	8339386800.606 [± 730340037.273]
GA-CX-R-ST	**2020734590.928** **[± 2393872683.029]**	8122230062.891 [± 217213422.096]	**2168923672.647** **[± 8893693282.361]**	8310330051.728 [± 711071173.232]

Tabel 4.9. Nilai Failr rata-rata untuk delapan belas penjadwal berbasis GA dalam skenario aman [sd], (s.d. = standar deviasi)

	Static		Dynamic	
Strategy	**Medium**	**Very Large**	**Medium**	**Very Large**
GA-PMX-M-SS	44.67 [± 7.9]	49.40 [± 8.9]	43.81 [± 8.12]	47.12 [± 6.05]
GA-PMX-M-EG	47.83 [± 9.82]	47.01 [± 5.54]	44.80 [± 8.19]	48.63 [± 7.50]
GA-PMX-M-ST	43.547 [± 7.91]	47.95 [± 7.56]	40.77 [± 5.22]	48.37 [± 6.46]
GA-PMX-S-SS	42.99 [± 9.92]	48.68 [± 6.75]	44.04 [± 9.79]	47.22 [± 8.56]
GA-PMX-S-EG	43.48 [± 7.61]	49.67 [± 6.09]	44.16 [± 8.53]	47.66 [± 7.99]
GA-PMX-S-ST	41.82 [± 7.98]	48.93 [± 9.23]	43.26 [± 5.20]	49.11 [± 7.73]
GA-PMX-R-SS	43.09 [± 9.37]	48.55 [± 5.70]	43.90 [± 9.43]	48.74 [± 5.65]
GA-PMX-R-EG	43.07 [± 8.37]	46.47 [± 5.41]	43.69 [± 5.92]	48.93 [± 7.48]
GA-PMX-R-ST	42.96 [± 5.53]	47.39 [± 6.88]	43.06 [± 9.72]	48.74 [± 5.52]
GA-CX-M-SS	42.78 [± 7.15]	47.93 [± 7.25]	42.53 [± 9.25]	48.10 [± 6.93]
GA-CX-M-EG	41.96 [± 6.54]	48.39 [± 8.22]	42.94 [± 7.3]	49.11 [± 9.57]
GA-CX-M-ST	41.64 [± 9.77]	47.91 [± 8.64]	42.12 [± 6.79]	48.41 [± 7.24]
GA-CX-S-SS	42.54 [± 5.66]	45.65 [± 7.71]	43.28 [± 6.15]	48.65 [± 9.69]
GA-CX-S-EG	42.84 [± 9.55]	44.26 [± 9.30]	40.45 [± 9.71]	48.22 [± 9.52]
GA-CX-S-ST	38.66 [± 9.59]	46.08 [± 7.58]	39.47 [± 6.5]	48.27 [± 8.98]
GA-CX-R-SS	39. 13 [± 8.57]	43.34 [± 9.78]	41.67 [± 8.15]	46.12 [± 7.18]
GA-CX-R-EG	39.72 [± 6.56]	43.75 [± 6.77]	41.03 [± 9.46]	44.97 [± 7.84]

GA-CX-R-ST	**37.24 42.20**	**38.33 43.05**

Tabel 4.10. Nilai Failr rata-rata untuk delapan belas penjadwal berbasis GA dalam skenario berisiko [s.d.], (s.d. = standar deviasi)

Strategy	Static		Dynamic	
	Medium	**Very Large**	**Medium**	**Very Large**
GA-PMX-M-SS	44.29 [± 6.82]	51.99 [± 6.43]	45.66 [± 5.29]	53.65 [± 9.52]
GA-PMX-M-EG	44.68 [± 7.95]	52.07 [± 6.68]	45.50 [± 6.02]	53.72 [± 9.92]
GA-PMX-M-ST	44.92 [± 7.82]	51.80 [± 6.83]	45.49 [± 9.94]	53.42 [± 4.35]
GA-PMX-S-SS	44.02 [± 9.43]	51.88 [± 7.39]	45.39 [± 6.68]	53.21 [± 6.83]
GA-PMX-S-EG	43.28 [± 10.02]	51.99 [± 7.37]	45.12 [± 8.93]	53.09 [± 8.87]
GA-PMX-S-ST	43.94 [± 8.21]	51.77 [± 5.36]	44.98 [± 5.83]	52.94 [± 7.66]
GA-PMX-R-SS	43.91 [± 8.99]	51.18 [± 5.84]	44.64 [± 5.67]	52.54 [± 5.87]
GA-PMX-R-EG	43.99 [± 6.29]	51.99 [± 9.34]	44.83 [± 7.69]	52.53 [± 10.02]
GA-PMX-R-ST	43.64 [± 8.55]	52.03 [± 9.64]	43.93 [± 9.34]	52.01 [± 5.38]
GA-CX-M-SS	44.01 [± 9.22]	51.93 [± 8.33]	43.14 [± 7.74]	51.88 [± 9.95]
GA-CX-M-EG	43.95 [± 8.69]	51.29 [± 7.36]	43.02 [± 8.17]	52.09 [± 5.73]
GA-CX-M-ST	42.99 [± 10.13]	50.92 [± 8.63]	42.85 [± 8.70]	51.98 [± 7.92]
GA-CX-S-SS	42.22 [± 5.98]	50.78 [± 5.83]	42.99 [± 6.89]	51.96 [± 9.67]
GA-CX-S-EG	41.97 [± 7.03]	49.63 [± 9.92]	42.93 [± 8.28]	51.62 [± 8.32]
GA-CX-S-ST	41.08 [± 7.55]	50.20 [± 10.22]	43.03 [± 9.53]	51.93 [± 8.76]
GA-CX-R-SS	41.85 [± 8.06]	49.83 [± 9.53]	42.73 [± 6.72]	51.01 [± 7.47]
GA-CX-R-EG	40.59 [± 6.92]	49.21 [± 9.02]	4240 [± 7.83]	50. 39 [± 9.93]
GA-CX-R-ST	**39.88**	**47.92**	41.92	**50.38**

Tabel 4.11. Rata-rata nilai Im(E) untuk delapan belas penjadwal berbasis GA dalam skenario aman [±s.d.], (s.d. = standar deviasi)

Strategy	Static		Dynamic	
	Medium	**Very Large**	**Medium**	**Very Large**
GA-PMX-M-SS	17.65 [± 2.49]	13.95 [± 2.94]	15.99 [± 4.1]	12.01 [± 1.89]

GA-PMX-M-EG	17.99 15.24 [± 3.55] [± 1.44]	16.44 13.10 [± 2.39] [± 1.82]
GA-PMX-M-ST	18.10 15.37 [± 2.18] [± 2.07]	16.35 12.95 [± 2.47] [± 1.93]
GA-PMX-S-SS	18.03 15.61 [± 3.09] [± 2.40]	16.31 13.06 [± 3.59] [± 5.50]
GA-PMX-S-EG	18.73 15.85 [± 4.26] [± 3.62]	16.32 13.80 [± 3.45] [± 2.50]
GA-PMX-S-ST	19.67 16.10 [± 4.98] [± 4.51]	16.25 13.73 [± 3.11] [± 2.78]
GA-PMX-R-SS	19.65 16.49 [± 2.40] [± 2.53]	16.86 13.38 [± 3.7] [± 2.74]
GA-PMX-R-EG	19.56 16.36 [± 3.73] [± 3.26]	17.28 15.11 [± 4.01] [± 4.28]
GA-PMX-R-ST	19.76 18.63 [± 3.75] [± 2.29]	19.04 15.77 [± 5.99 [± 3.74]
GA-CX-M-SS	20.564 19.53 [± 3.85] [± 2.82]	18.63 15.94 [± 1.99] [± 3.74]
GA-CX-M-EG	21.23 20.43 [± 5.89] [± 2.12]	18.36 17.98 [± 6.04] [± 3.12]
GA-CX-M-ST	23.84 22.12 [± 3.98] [± 4.39]	18.37 16.28 [± 5.18] [± 4.02]
GA-CX-S-SS	25.70 20.83 [± 6.33] [± 7.80]	19.27 19.02 [± 4.85] [± 6.00]
GA-CX-S-EG	27.72 21.90 [± 2.32] [± 7.34]	20.82 19.99 [± 4.29] [± 4.65]
GA-CX-S-ST	26.87 21.35 [± 6.83] [± 5.25]	20.37 20.23 [± 5.77] [± 7.06]
GA-CX-R-SS	26.99 21.02 [± 5.89] [± 5.32]	22.87 20.39 [± 4.19] [± 6.37]
GA-CX-R-EG	28.14 21.63 [± 4.22] [± 5.30]	27.91 20.19 [± 8.01] [± 5.83]
GA-CX-R-ST	**35.38 28.15**	**30.93 32.05**

Tabel 4.12. Rata-rata nilai Im(E) untuk delapan belas penjadwal berbasis GA dalam skenario berisiko [±s.d.], (s.d. = standar deviasi)

	Static	Dynamic
Strategy	**Medium Very Large**	**Medium Very Large**
GA-PMX-M-SS	10.43 9.98 [± 1.59] [± 1.43]	8.12 8.06 [± 1.41] [± 1.40]
GA-PMX-M-EG	11.13 10.23 [± 1.12] [± 1.43]	8.42 8.66 [± 1.70] [± 1.66]
GA-PMX-M-ST	11.07 10.08 [± 0.99] [± 1.25]	8.34 8.60 [± 1.46] [± 1.69]
GA-PMX-S-SS	13.63 12.62 [± 1.82] [± 1.418]	11.60 11.05 [± 1.65] [± 1.85]
GA-PMX-S-EG	14.86 12.92 [± 1.84] [± 1.49]	11.93 11.31 [± 1.45] [± 1.57]

GA-PMX-S-ST	15.37 13.71 [± 2.44] [± 4.43]	12.37 11.325 [± 1.781] [± 1.75]
GA-PMX-R-SS	15.96 14.29 [± 2.55] [± 2.90]	14.33 13.47 [± 3.51] [± 3.74]
GA-PMX-R-EG	16.95 15.44 [± 1.67] [± 2.83]	15.28 14.03 [± 3.52] [± 4.70]
GA-PMX-R-ST	18.41 17.83 [± 3.76] [± 2.42]	15.34 15.08 [± 4.21] [± 3.92]
GA-CX-M-SS	18.41 16.99 [± 2.62] [± 2.571]	14.22 13.00 [± 3.82] [± 2.78]
GA-CX-M-EG	18.25 17.04 [± 3.97] [± 3.05	14.83 13.40 [± 2.21] [± 3.59]
GA-CX-M-ST	19.03 17.01 [± 2.94] [± 3.28]	15.01 14.21 [± 3.64] [± 1.85]
GA-CX-S-SS	19.26 18.83 [± 2.78] [± 2.77]	17.45 15.21 [± 5.58] [± 4.73]
GA-CX-S-EG	19.74 18.36 [± 1.87] [± 3.29]	18.91 16.57 [± 3.22] [± 2.81]
GA-CX-S-ST	19.34 18.35 [± 2.13] [± 2.39]	18.78 16.98 [± 3.52] [± 3.94]
GA-CX-R-SS	20.66 18.98 [± 2.98] [± 1.97]	19.96 17.91 [± 2.99] [± 3.82]
GA-CX-R-EG	20.17 19.22 [± 3.65] [± 4.29]	20.06 17.98 [± 3.02] [± 5.00]
GA-CX-R-ST	**21.77 20.20**	**21.33 18.03**

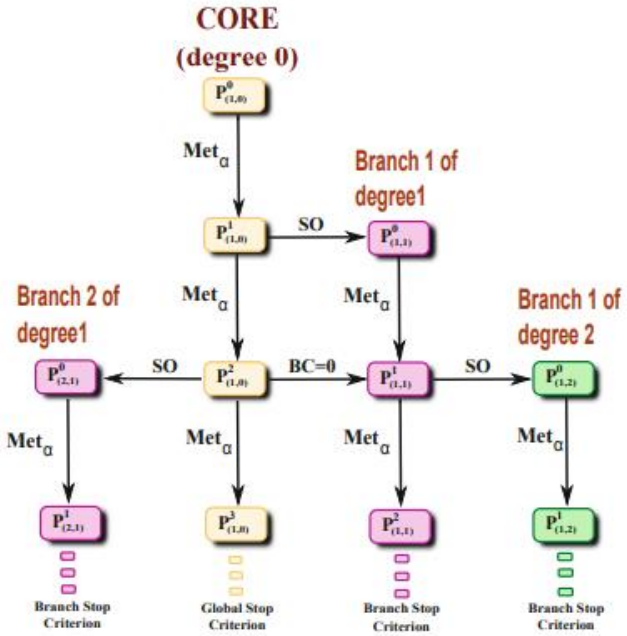
Dengan meringkas hasil untuk semua fungsi tujuan, algoritma GA-CX-R-ST adalah yang terbaik dalam meminimalkan semua tujuan yang dipertimbangkan. Juga harus diperhatikan bahwa dalam mode berisiko hampir semua hasil yang dicapai lebih buruk daripada dalam mode aman. Ini berarti bahwa dalam hal mengabaikan semua kondisi keamanan, tingkat kegagalan yang tinggi meningkatkan energi yang digunakan oleh sistem, dan tentu saja memperpanjang tenggat waktu untuk menyelesaikan tugas-tugas tertentu dan seluruh jadwal.

4.7 PENJADWAL JARINGAN GENETIK MULTI-POPULASI

Karena ukuran grid dan dinamika lingkungan grid, eksplorasi lanskap optimasi yang dihasilkan untuk masalah penjadwalan grid tetap merupakan tugas yang menantang untuk penjadwal grid berbasis genetik sederhana, terutama dalam kasus banyak kriteria penjadwalan yang berbeda (bukan hanya Makespan dan Flowtime). Dalam kasus seperti itu, keefektifan algoritme dapat ditingkatkan dengan menjalankannya secara bersamaan pada banyak populasi sebagai proses paralel. Semua GA populasi tunggal disajikan di bagian 4.5 dapat digunakan sebagai mekanisme genetik utama dalam meta-heuristik genetik multi-populasi. Dalam karya ini, kami mempertimbangkan dua penjadwal jaringan ketahanan risiko berbasis genetik multi-populasi berikut:

- HGS-Sched (R) – penjadwal HGS-Sched hirarkis multi-populasi yang bekerja dalam mode berisiko;
- HGS-Sched (S) – penjadwal HGS-Sched hirarkis multi-populasi yang bekerja dalam mode aman;
- IGA (R) – penjadwal GA pulau multi-populasi yang bekerja dalam mode berisiko;
- IGA (S) – penjadwal GA pulau multi-populasi yang bekerja dalam mode aman.

Ide utama dari Hierarchic Genetic Scheduler (HGS-Sched) didasarkan pada konsep pencarian bersamaan di lingkungan grid dengan pelaksanaan banyak proses evolusi yang bergantung. Strategi ini dimodelkan sebagai pohon keputusan multi-level. Proses pencarian di HGS-Sched diinisialisasi dengan mengaktifkan penjadwal dengan akurasi pencarian rendah. Penjadwal ini adalah modul utama dari keseluruhan strategi dan bertanggung jawab atas "manajemen" struktur umum pohon5 dan eksplorasi wilayah baru dan yang belum dikenal dalam domain pengoptimalan. Keakuratan pencarian di cabang-cabang HGS-Sched ditentukan oleh parameter derajat, dan tergantung pada tingkat mutasi dan ukuran populasi, yang dapat berbeda di cabang-cabang dengan derajat yang berbeda. Gambar 4.4 menggambarkan representasi grafis sederhana dari struktur 3 level HGS-Sched. Cabang-cabang baru ditumbuhkan dengan menggunakan prosedur bertunas SO setelah pelaksanaan proses evolusi generasi-α (metaepoch α-periodik Metα). Perpanjangan pohon diarahkan dengan menggunakan Branch Comparison (BC).



Gambar 4.4. 3 tingkat struktur pohon HGS-Sched

Algoritma Genetika Pulau (IGA) adalah teknik GA paralel yang terkenal. Populasi awal (biasanya besar) dibagi menjadi beberapa sub-populasi, "pulau" atau "demes", di mana GA populasi tunggal dengan konfigurasi parameter dan operator yang identik diaktifkan (satu algoritme untuk setiap deme). Setelah jumlah iterasi tetap, dilambangkan sebagai itd, prosedur migrasi diaktifkan. Ini memungkinkan pertukaran parsial (biasanya sesuai dengan topologi cincin standar) dari individu di antara pulau-pulau. Jumlah relatif individu yang

bermigrasi, diwakili oleh mig, adalah parameter global algoritma yang umumnya dikenal sebagai tingkat migrasi, dan dihitung sebagai:

$$mig = \frac{m_{deme}}{deme} \cdot 100\% \qquad (4.45)$$

dimana deme adalah ukuran subpopulasi di IGA dan mdeme adalah jumlah individu yang bermigrasi di setiap deme. Prosedur migrasi diulang setelah setiap eksekusi iterasi algoritma genetika di setiap sub-populasi.

Algoritma GA – CX - R – ST (R) dan GA – CX – R – ST (S), dipilih sebagai penjadwal populasi tunggal yang paling efektif di bab 4.6, telah digunakan sebagai mekanisme genetik utama dalam semua implementasi HGS-Sched (di semua jenis cabang) dan IGA.

Analisis Empiris

Pada bagian ini kami menyajikan hasil evaluasi empiris dari dua jenis penjadwal genetik multipopulasi dan membandingkannya dengan hasil yang dicapai oleh algoritma GA – CX – R – ST (R) dan GA – CX – R – ST(S). Serupa dengan eksperimen sebelumnya, kami telah menggunakan simulator jaringan Sim-G-Batch dengan konfigurasi yang sama seperti yang ditentukan di Sec. 4.6. Konfigurasi parameter kunci untuk kedua implementasi algoritma GA – CX – R – ST (x) sama seperti dalam eksperimen yang disediakan untuk semua jenis penjadwal populasi tunggal (lihat Tabel 4.4), parameter untuk IGA dan Green-HGS-Meta-heuristik terjadwal disajikan pada Tabel 4.13 dan 4.14.

Hasil

Setiap percobaan diulang 30 kali di bawah konfigurasi operator dan parameter yang sama. Kotak-plot analisis statistik dari nilai empat ukuran kinerja penjadwal yang dirata-ratakan dalam 30 putaran simulator disajikan pada Gambar 4.5–4.8.

Dalam kasus Makespan, Flowtime dan Failr, hasil terbaik dicapai oleh algoritma HGS – Sched(S). Khususnya dalam grid dinamis, perbedaan hasil yang dihasilkan oleh penjadwal ini dan meta-heuristik lainnya sangat signifikan. Dalam kasus grid "Sedang", algoritma GA – CX – R – ST(S) populasi tunggal lebih efektif dalam minimisasi Makespan daripada implementasi yang aman dari model pulau, yang menunjukkan bahwa dalam kasus ini solusi terbaik yang ditemukan dari masalah mungkin terletak sangat dekat satu sama lain, dan prosedur pulau tidak meningkatkan kualitas pencarian mekanisme penjadwalan populasi tunggal yang diterapkan di setiap deme IGA. Dalam semua skenario penjadwalan, implementasi algoritma yang aman dari kelas yang sama lebih efektif daripada implementasi berisiko untuk ketiga kriteria penjadwalan yang disebutkan di atas.

Situasinya sedikit berbeda dalam hal minimisasi konsumsi energi. Dalam hal ini implementasi berisiko dari algoritme HGS-Sched dan IGA mencapai nilai terbaik (tertinggi) dari tingkat peningkatan energi masing-masing di jaringan “Sedang” dan “Sangat besar”. Hal ini dapat mengarah pada kesimpulan bahwa menurunkan catu daya sistem dalam kasus tersebut tidak meningkatkan nilai Makespan dan Flowtime, yang cukup besar untuk kedua penjadwal tersebut, tetapi memungkinkan untuk mengurangi penggunaan energi secara signifikan.

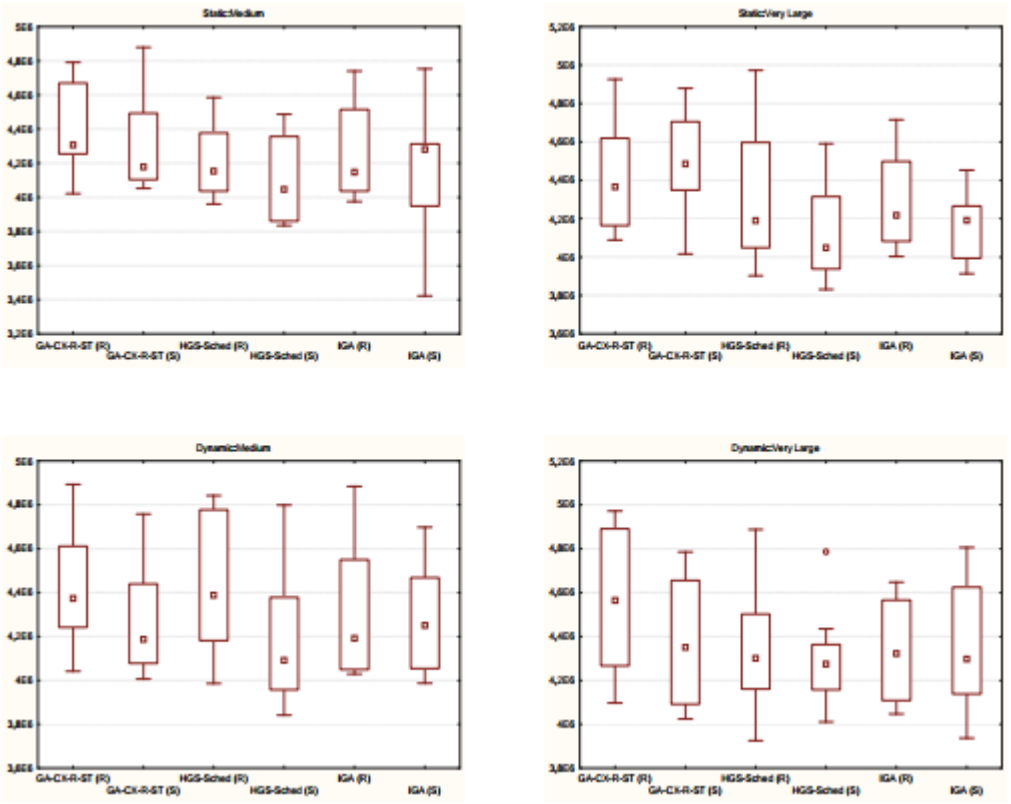
Dapat diamati bahwa HGS-Sched (S) adalah penjadwal yang paling stabil di semua skenario yang dipertimbangkan. Perbedaan antara nilai minimal dan maksimal, dan kuantil pertama dan ketiga adalah yang terendah untuk meta-heuristik ini.

Tabel 4.13. Pengaturan HGS-Sched untuk tolok ukur statis dan dinamis

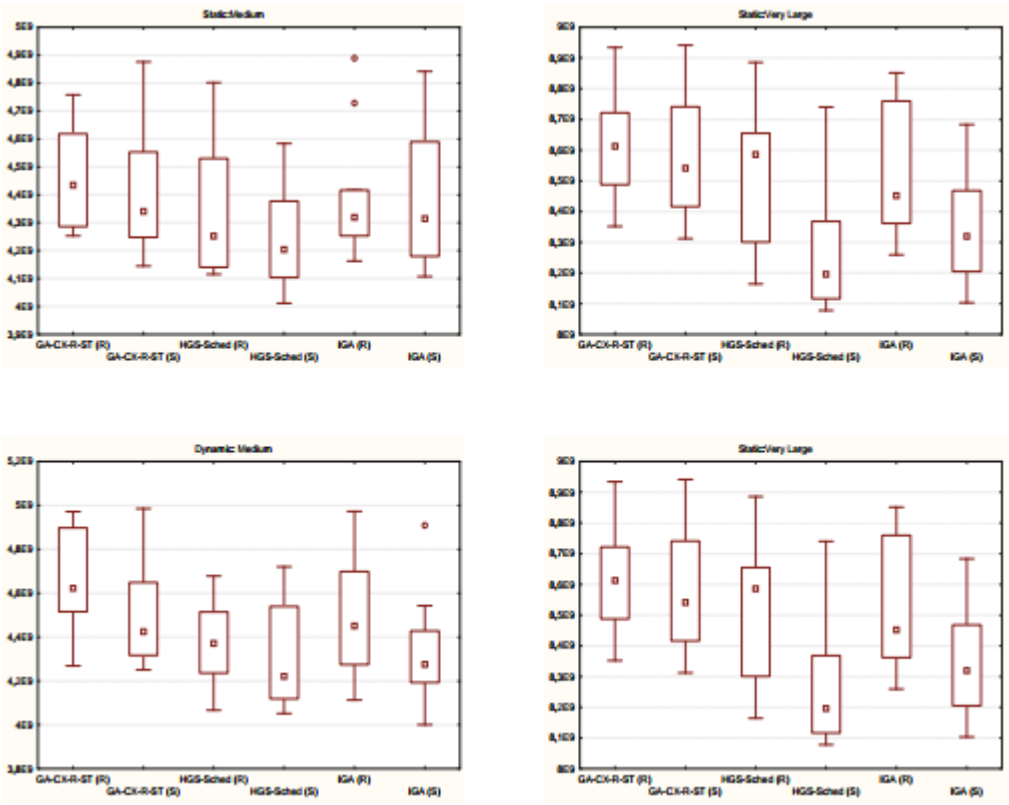
Parameter	
period_of_metaepoch	$20 * n$
nb_of_metaepochs	10
degrees of branches (t)	0 and 1
population size in the core	$3 * (\lceil 4 * (\log_2 n - 1)/(11.8) \rceil)$
population size in the sprouted branches (*b_pop_size*)	$(\lceil (4 * (\log_2 n - 1)/(11.8) \rceil)$
intermediate pop. in the core	*abs*((*r_pop_size*)/3)
intermediate pop. in the sprouted branch	*abs*((*b_pop_size*)/3)
cross probab.	0.9
mutation probab. in core	0.4
mutation probab. in the sprouted branches	0.2
max_time_to_spend	40 secs (*static*) / 70 secs (*dynamic*)

Tabel 4.14. Konfigurasi algoritma IGA

Parameter	
it_d	$20 * n$
mig	5 %
number of islands (demes)	10
cross probab.	1.0
mutation probab.	0.2
max_time_to_spend	40 secs (*static*) / 70 secs (*dynamic*)



Gambar 4.5. Kotak-plot hasil untuk Makespan dalam skenario statis dan dinamis



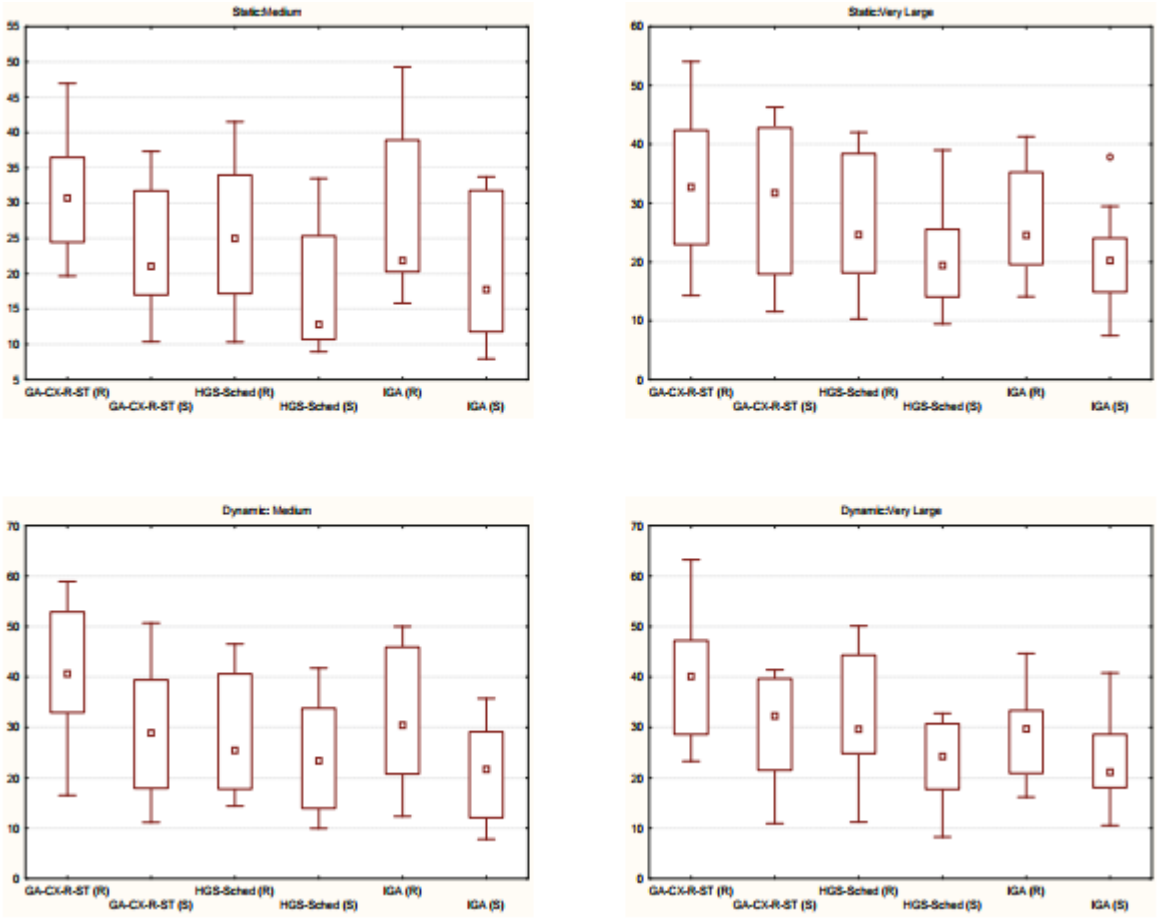
Gambar 4.6. Kotak-plot hasil Mean Flowtime dalam skenario statis dan dinamis

4.8 PEKERJAAN TERKAIT

Masalah keamanan dan keandalan sumber daya telah dipelajari secara intensif di banyak publikasi selama beberapa tahun terakhir. Namun, banyak pendekatan penjadwalan terkenal untuk komputasi grid sebagian besar mengabaikan faktor keamanan, dengan hanya segelintir pengecualian.

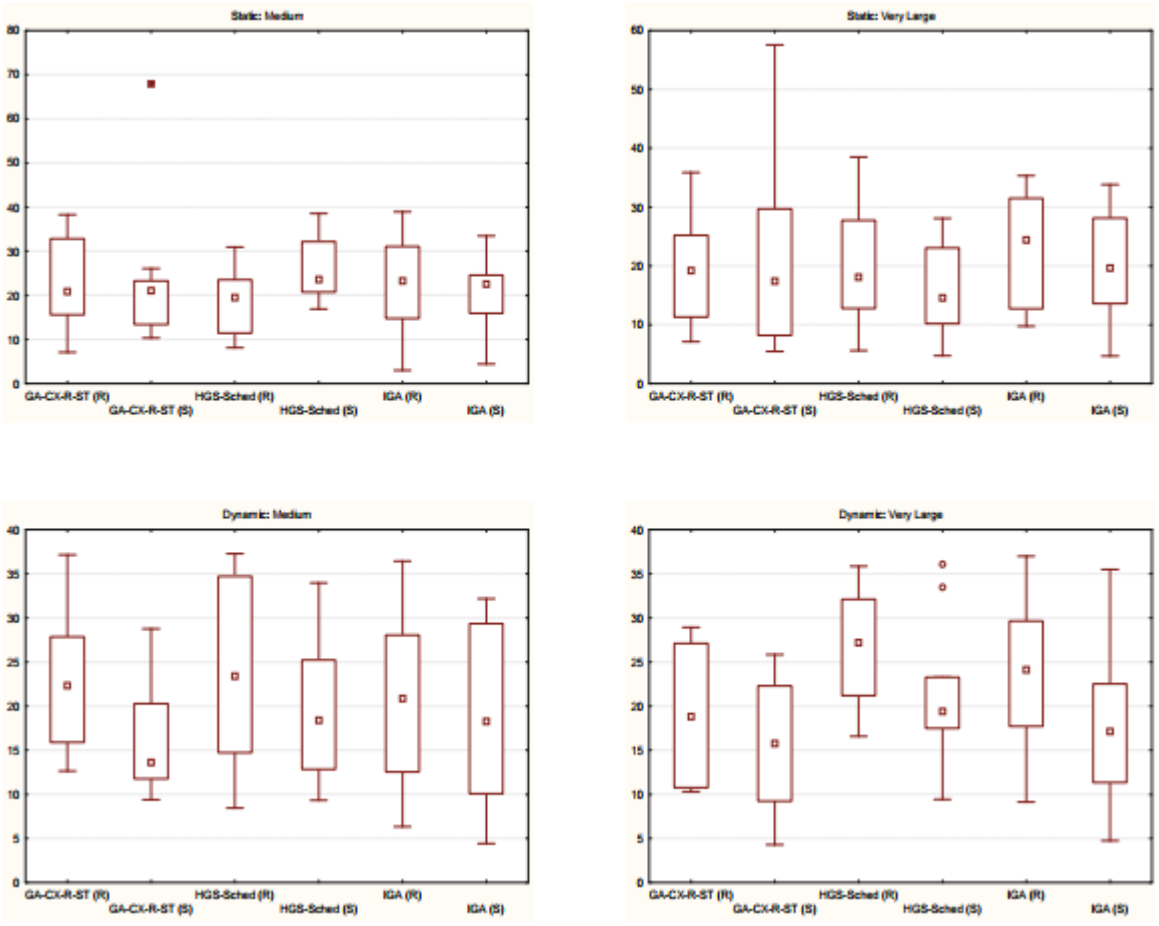
Banyak upaya telah dilakukan untuk mengembangkan modul manajemen cerdas dalam sistem grid untuk mengontrol akses layanan melalui otentikasi dalam penjadwalan online, atau untuk memantau pelaksanaan aplikasi grid dan mendeteksi kegagalan sumber daya karena pembatasan keamanan. Dalam mengembangkan sebuah model, di mana pekerjaan direplikasi di beberapa situs jaringan untuk meningkatkan kemungkinan pemenuhan persyaratan keamanan dan eksekusi pekerjaan yang berhasil.

Meskipun penjadwalan sadar-keamanan dalam grid telah dipelajari dengan baik, tidak banyak contoh aplikasi yang berhasil dari meta-heuristik berbasis genetik untuk memecahkan masalah ini. Di sebagian besar publikasi dalam keamanan domain dan mekanisme aborsi tugas biasanya diimplementasikan sebagai prosedur eksternal yang terpisah dari inti sistem penjadwalan.



Gambar 4.7. Kotak-plot hasil untuk parameter Failr dalam skenario statis dan dinamis

Salah satu pendekatan sadar keamanan yang menjanjikan dalam penjadwalan grid didasarkan pada model teoretis permainan. Penulis mengembangkan model prediksi mobilitas simpul untuk sistem jaringan seluler dan menggunakan strategi penetapan harga wajar yang telah ditentukan sebelumnya untuk mengamankan akses simpul jaringan seluler. Dalam semua model yang disebutkan di atas, keputusan akhir tentang alokasi aman tugas ke sumber daya dibuat oleh pengguna jaringan yang tidak bekerja sama satu sama lain. Biaya eksekusi tugas yang tahan risiko ditafsirkan sebagai fungsi biaya pengguna, yang ditentukan sebagai tujuan penjadwalan dan diminimalkan selama permainan. Kelemahan utama dari pendekatan ini adalah kompleksitas komputasinya. Dalam banyak kasus, game disediakan pada level grid yang berbeda dan untuk menentukan mekanisme sinkronisasi yang efektif adalah tugas yang menantang.



Gambar 4.8. Kotak-plot hasil Tingkat Peningkatan Energi dalam skenario statis dan dinamis

Volume penelitian yang signifikan telah dilakukan dalam domain manajemen sumber daya sadar energi dalam sistem komputasi terdistribusi skala besar modern. Beberapa karya penelitian telah menggunakan model dan pendekatan serupa yang terkait dengan komputasi hijau dalam sistem terdistribusi skala besar, seperti proporsionalitas energi, komputasi sadar memori, komputasi intensif data, hemat energi, dan grid penjadwalan.

Masih belum ada keluarga besar penjadwal berbasis genetik yang sadar energi di lingkungan grid dan cloud. Di sebagian besar pendekatan DVFS, penjadwalan telah didefinisikan sebagai masalah load balancing klasik atau dinamis. Dalam kasus seperti itu, pemrograman linier, dinamis, dan sasaran adalah teknik pengoptimalan utama.

4.9 KESIMPULAN

Alasan utama di balik kompleksitas penjadwalan grid multi-kriteria adalah bahwa masalah ini terdiri dari beberapa komponen yang saling berhubungan (kriteria, sub-masalah), yang dapat membuat banyak pendekatan standar menjadi tidak efektif. Bahkan jika algoritme yang tepat dan efisien untuk memecahkan komponen atau aspek tertentu dari masalah keseluruhan sudah diketahui, algoritme ini hanya menghasilkan solusi untuk sub-masalah, dan tetap menjadi pertanyaan terbuka bagaimana mengintegrasikan solusi parsial ini untuk mencapai solusi optimal global. bungkam. Meta-heuristik, karena ketangguhan dan

skalabilitasnya yang tinggi, mampu menangani berbagai atribut dan kriteria penjadwalan yang terkadang saling bertentangan.

Analisis empiris komprehensif yang disajikan dalam bab ini menunjukkan efektivitas tinggi dari metaheuristik berbasis genetik populasi tunggal dan multi populasi dalam optimalisasi tujuan penjadwalan jaringan konvensional, yaitu Makespan dan Flowtime, serta kriteria baru seperti energi kumulatif yang dikonsumsi. dalam sistem jaringan dan masalah keamanan. Kriteria tambahan ini biasanya dianggap sebagai masalah pengoptimalan yang terpisah. Di sini, dalam pendekatan kami, mereka disematkan dalam model penjadwalan yang diusulkan. Skenario grid yang disimulasikan dalam kasus seperti itu dapat menggambarkan sistem realistis dengan lebih baik, di mana sejumlah besar variabel, banyak tujuan, kendala, dan aturan bisnis, semuanya berkontribusi dalam berbagai cara harus dianalisis.

Semua model yang disajikan dalam bab ini sebenarnya tidak terbatas hanya pada sistem grid. Mereka dapat dengan mudah diadaptasi ke lingkungan cloud, di mana kesadaran keamanan dan manajemen daya yang cerdas adalah masalah penelitian yang paling populer.

BAB 5
PENJADWALAN KONSUMSI DAYA MENGGUNAKAN ALGORITMA

Dua tantangan khas dalam penjadwalan tugas sadar energi adalah (1) meminimalkan konsumsi energi dengan batasan waktu eksekusi dan (2) meminimalkan waktu eksekusi tugas dengan batasan konsumsi energi. Bab ini berfokus pada tantangan selanjutnya. Sangat penting untuk menjadwalkan tugas secara efisien di perangkat seluler dan jaringan sensor yang memiliki daya terbatas. Tujuannya adalah untuk secara kooperatif menyelesaikan serangkaian tugas di antara beragam sumber daya komputasi dengan energi yang diberikan. Algoritma penjadwalan tradisional, seperti penjadwalan daftar, tidak terlalu efisien untuk masalah penjadwalan ini. Metode pengoptimalan pemrograman linier tidak cocok karena merupakan masalah non-linier. Algoritma genetik yang ditingkatkan dapat memecahkan masalah ini secara efektif.

5.1 PENDAHULUAN

Perangkat komputasi kecil, cepat, dan portabel, seperti komputer laptop nirkabel, ponsel, iPad, iCloud, sangat populer. Mereka semua bergantung pada energi perangkat yang terbatas. Penting untuk membuatnya lebih hemat daya.

Teknologi *cloud computing* berbasis jaringan memberikan kemampuan komputasi yang lebih canggih. Mereka dapat dirancang sebagai beberapa komputer yang terhubung secara longgar melalui internet, atau banyak komputer yang dihosting di pusat data. Pusat data lebih efektif dalam menyediakan layanan komputasi yang kompleks. Ini dapat secara efisien menyediakan berbagai layanan komputasi yang diminta oleh perangkat komputasi portabel, menyelesaikan transaksi bisnis yang rumit, dan melakukan perhitungan ilmiah.

Prakiraan cuaca adalah contoh yang baik tentang bagaimana komputer pribadi dan pusat data bekerja sama. Prakiraan cuaca adalah proses yang sangat rumit. Dibutuhkan banyak pengamatan cuaca, pemrosesan pencitraan satelit, simulasi cuaca, dan penjelasan ahli meteorologi. Ini menuntut sejumlah besar daya komputasi untuk menyelesaikan pekerjaan dengan cepat. Komputer besar yang kuat di pusat data digunakan untuk tugas semacam ini. Hasil prakiraan cuaca disiarkan ke perangkat komputer pribadi yang digunakan oleh banyak konsumen.

Komputer mengonsumsi energi dalam dua cara umum, konsumsi terkait komputasi langsung dan tidak langsung. Energi yang dikonsumsi oleh perangkat pendukung, seperti AC di data center, merupakan konsumsi energi tidak langsung. Energi yang digunakan oleh komputer adalah konsumsi energi langsung. Bersama-sama, konsumsi energi terkait komputasi kira-kira setara dengan konsumsi energi industri penerbangan. Ini menyumbang 2% CO_2 antropogenik dari bagian konsumsi energinya.

Sebuah pusat komputer dapat menampung 10.000 atau 150.000 server. Pusat data mega ini dapat mendukung operasi harian banyak perusahaan besar, melakukan banyak transaksi e-commerce, melakukan penelitian ilmiah skala besar, dan memberikan layanan kepada banyak klien lainnya. Pusat data ini menggunakan energi dalam jumlah besar. Energi

yang digunakan oleh server dan pusat data AS sangat signifikan. Diperkirakan mereka mengonsumsi sekitar 61 miliar kilowatt-jam (kWh) pada tahun 2006 (1,5% dari total konsumsi listrik AS) dengan total biaya listrik sekitar 4,5 miliar USD. Jika tren ini berlanjut, permintaan ini akan meningkat menjadi 12 gigaWatt (GW) pada tahun 2011. Ini akan membutuhkan tambahan 10 pembangkit.

Energi hijau adalah energi yang dihasilkan dari sumber terbarukan, seperti matahari, angin, panas bumi, biomassa, dan hidro kecil. Mereka adalah sumber terbarukan dan lebih ramah lingkungan daripada metode pembangkitan listrik tradisional. Mereka memancarkan sedikit atau tidak ada polusi udara dan tidak meninggalkan limbah radioaktif seperti pembangkit listrik tenaga nuklir. Yang terpenting, mereka secara alami diisi ulang oleh bumi dan matahari.

Energi coklat adalah kekuatan yang dihasilkan dari teknologi bermusuhan lingkungan. Sebagian besar listrik yang dikonsumsi di Amerika Serikat berasal dari pembangkit batu bara, nuklir, hidro besar, dan gas alam. Ini adalah satu-satunya sumber polusi udara terbesar di Amerika Serikat, yang berkontribusi terhadap kabut asap dan hujan asam. Ini adalah penyumbang tunggal terbesar gas perubahan iklim global, termasuk karbon dioksida dan nitrogen oksida.

Sebagian besar energi yang kita konsumsi saat ini adalah energi bermusuhan lingkungan yang tidak dapat diperbarui. Pada tahun 2006, energi hijau hanya menyumbang 7% dari total pasokan energi AS. Pembakaran minyak bumi, batubara dan gas alam menghasilkan 86% dari total pasokan energi. Nafsu kita akan lebih banyak energi tumbuh lebih cepat dari sebelumnya.

Banyak proyek penelitian telah dilakukan untuk meningkatkan efisiensi energi pusat data, seperti memperbaiki desain pusat data, memperbaiki peralatan, dan memperbaiki AC. Mereka berfokus pada pengurangan konsumsi energi dan meningkatkan efisiensi perangkat pendukung.

Meningkatkan efisiensi penggunaan energi dan mengurangi penggunaan energi adalah dua pendekatan utama konservasi energi dan komputasi hijau. Penjadwalan tugas yang efisien di pusat data adalah salah satu topik penelitian yang dapat menghasilkan pengembalian yang signifikan. Dengan penjadwalan tugas yang dioptimalkan, komputer dapat menyelesaikan tugas menggunakan lebih sedikit energi. Ini juga mengurangi konsumsi energi dari perangkat pendukung. Gabungan penghematan energi dari penjadwalan tugas yang efisien di pusat data yang besar dapat menjadi signifikan.

Dari perspektif komputasi hijau, penjadwalan tugas yang efisien dapat didefinisikan sebagai meminimalkan konsumsi energi dengan batasan panjang jadwal atau meminimalkan panjang jadwal dengan batasan konsumsi energi. Tujuan dari masalah pertama adalah menggunakan energi paling sedikit untuk menyelesaikan semua tugas dalam jangka waktu tertentu. Ini digunakan terutama di lingkungan pemrosesan waktu nyata. Masalah kedua adalah menyelesaikan tugas secepat mungkin di bawah batasan energi yang diberikan. Tujuannya adalah untuk menggunakan energi secara efisien dan memiliki kegunaan yang besar dalam komputasi seluler, jaringan sensor, dll. Ada banyak algoritma penjadwalan tugas hemat energi untuk beberapa komputer heterogen. Algoritma heuristik dapat menemukan

solusi yang baik di antara semua solusi yang mungkin. Mereka cenderung menyelesaikan pencarian dengan sangat cepat. Mereka mungkin tidak selalu menemukan solusi terbaik tetapi menjamin solusi optimal lokal. Algoritma ini biasanya dapat menemukan yang mendekati solusi optimal.

Algoritma pencarian yang terinspirasi bio, cari ruang dengan mensimulasikan proses alam. Algoritme tipikal adalah Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), dll. Mereka dapat menemukan solusi optimal atau mendekati optimal. Mereka lebih lambat dari algoritma heuristik pada arsitektur komputer saat ini. Algoritma hibrid mencoba menggabungkan atribut yang lebih baik dari berbagai algoritma.

Kami memecahkan masalah konsumsi energi meminimalkan pertama menggunakan GA yang disempurnakan. Kami memperkenalkan konsep Shadow Price ke dalam GA dan mencapai hasil yang jauh lebih baik. Dalam bab ini, kami memperkenalkan Shadow Price yang ditingkatkan GA (S PGA) untuk memecahkan masalah kedua - meminimalkan waktu eksekusi tugas dengan batasan energi.

Sisa dari bab ini disusun sebagai berikut. Bagian 5.2 mendefinisikan masalah penjadwalan tugas yang dibatasi energi. Bagian 5.3 memperkenalkan algoritme penjadwalan tugas genetik terpandu harga bayangan baru kami. Bagian 5.4 menunjukkan dan menganalisis hasil percobaan. Bagian 5.5 menyimpulkan bab ini.

5.2 KONSUMSI DAYA MASALAH PENJADWALAN TUGAS TERKENDALA

Secara umum, jumlah daya yang digunakan perangkat listrik adalah produk dari memasok tegangan dan arus yang ditariknya. Energi yang dikonsumsi adalah produk dari daya dan waktu. Selain itu, kecepatan prosesor komputer bervariasi berdasarkan voltase yang diberikan. Dalam batas tertentu, prosesor bekerja lebih cepat dengan tegangan yang lebih tinggi. Dengan demikian, konsumsi daya sebuah prosesor secara langsung terkait dengan kecepatan kerjanya. Overclocking adalah salah satu teknik yang mempercepat prosesor dengan menaikkan voltase. Biaya peningkatan kecepatan ini adalah konsumsi energi yang lebih banyak. Tugas dapat diselesaikan lebih cepat dengan kecepatan yang lebih tinggi. Ini masalah pengoptimalan untuk mencapai keseimbangan antara energi dan waktu. Banyak komputer yang heterogen dan banyak tugas yang harus diselesaikan semakin memperumit tantangan.

Jika kita menghitung tugas *R* sebagai jumlah instruksi dan menjalankannya pada prosesor dengan kecepatan *S*, konsumsi daya untuk menyelesaikan tugas ini adalah $E = C \cdot R \cdot S^{\alpha-1}$. *C* adalah konstanta yang lebih besar dari 1, dan α adalah konstanta yang lebih besar dari 3. Perubahan kecepatan linier menghasilkan perubahan konsumsi energi secara eksponensial. Karena kecepatan adalah jumlah instruksi dari waktu ke waktu, konsumsi energi dapat direpresentasikan sebagai $E = C \cdot R(R/T)^{\alpha-1}$ dan T adalah waktu eksekusi tugas. Hitungan instruksi tugas memiliki dampak besar pada konsumsi energi juga.

Masalah yang akan dioptimalkan dapat didefinisikan sebagai: menggunakan waktu tersingkat untuk menyelesaikan *m* tugas pada *n* komputer heterogen dan total energi yang dikonsumsi kurang dari atau sama dengan *E.*

$$MinimizeT = max_{i \in n}(T_i) \quad (5.1)$$

tunduk pada

$$T_i = \sum_{j=1}^{k} \left(\frac{C_i \cdot \left(R_i^j\right)^{\alpha_i}}{E_i^j} \right)^{\frac{1}{\alpha_i - 1}} \quad (5.2)$$

$$E \geq \sum_{i=1}^{n} E_i \quad (5.3)$$

Dalam rumus fungsi tujuan (lihat Persamaan (5.1)), *Ti* adalah waktu eksekusi prosesor *i.* Persamaan. (5.2) adalah waktu eksekusi dari *k* tugas yang ditugaskan ke prosesor *i. Rj* mewakili jumlah instruksi dari tugas j yang ditugaskan ke prosesor *i. Ej* adalah energi yang digunakan, *Ci* dan *αi* adalah konstanta untuk prosesor *i, αi = 1 + 2/ϑi ≥ 3 dan 0 < ϑi ≤ 1*. Karena kecepatan umumnya digunakan dalam spesifikasi prosesor, Persamaan. (5.2) dapat disederhanakan menjadi ekspresi berikut:

$$T_i = \sum_{j=1}^{k} \frac{R_i^j}{S_i^j} \quad (5.4)$$

$$E_i = C_i \sum_{j=1}^{k} \left(R_i^j \left(S_i^j \right)^{\alpha_i - 1} \right) \quad (5.5)$$

$$X_i \leq S_i^j \leq Y_j \quad (5.6)$$

dimana *S j* adalah kecepatan eksekusi tugas *j* pada prosesor *i, Xi* dan *Yi* adalah kecepatan minimum dan maksimum dari prosesor *i, Rj* menyatakan jumlah instruksi tugas *j* yang diberikan pada prosesor *i.*

Tujuannya adalah untuk menemukan jadwal sehingga m tugas diselesaikan dalam waktu terpendek dan energi yang dikonsumsi berada dalam kendala *E*. Ada beberapa masalah suboptimalisasi dalam definisi, energi, tugas, dan kecepatan. Yang pertama adalah mendistribusikan batasan energi *E* secara optimal ke masing-masing prosesor *Ei.* Yang kedua adalah untuk memberikan tugas *R* secara optimal ke setiap prosesor. Dan yang terakhir adalah menentukan kecepatan lari optimal untuk setiap tugas yang diberikan ke prosesor. Semua tiga sub masalah terhubung. Menetapkan energi yang lebih tinggi ke prosesor memungkinkannya memproses lebih banyak tugas dalam waktu singkat. Kecepatan lari yang lebih tinggi membutuhkan lebih banyak energi. Karena tujuannya adalah untuk meminimalkan waktu berjalan terlama dari sebuah prosesor, semua prosesor harus bekerja sama dalam energi dan penetapan tugas. Ini adalah masalah optimalisasi kombinasional yang sangat kompleks.

Terbukti bahwa panjang jadwal diminimalkan ketika semua tugas yang ditugaskan ke prosesor dieksekusi dengan kekuatan yang sama. Untuk mencapai hasil terbaik, tugas yang diberikan ke prosesor yang sama harus dijalankan pada kecepatan yang sama karena daya menentukan kecepatan. Oleh karena itu, Persamaan. (5.2), (5.4), dan (5.5) dapat disederhanakan menjadi rumus berikut:

$$T_i = \left(\frac{C_i \cdot \left(\sum_{j=1}^{k} R_i^j \right)^{\alpha_i}}{E_i^j} \right)^{\frac{1}{\alpha_i - 1}} \tag{5.7}$$

$$T_i = \frac{\sum_{j=1}^{k} R_i^j}{S_i} \tag{5.8}$$

$$E_i = C_i \cdot \left(\sum_{j=1}^{k} R_i^j \right) \cdot (S_i)^{\alpha_i - 1} \tag{5.9}$$

Karena semua tugas berjalan dengan kecepatan yang sama pada prosesor yang sama dan tujuannya adalah menyelesaikan tugas secepat mungkin dengan energi yang ditetapkan untuk prosesor, kita dapat menggunakan Persamaan. (5.7) untuk menghitung waktu pelaksanaan, atau Persamaan. (5.10) untuk menghitung kecepatan optimal. Ini memecahkan masalah sub optimisasi ketiga. Apa yang perlu kita selesaikan sekarang adalah sub-masalah mendistribusikan konsumsi energi total yang diizinkan ke setiap prosesor dan menetapkan tugas kepada mereka sedemikian rupa sehingga waktu eksekusi minimal.

$$S_i = \left(\frac{E_i}{\left(C_i \cdot \left(\sum_{j=1}^{k} R_i^j \right) \right)} \right)^{\frac{1}{\alpha_i - 1}} \tag{5.10}$$

Ada dua fungsi objektif saat mengoptimalkan waktu eksekusi, meminimalkan waktu berjalan bersamaan pada semua prosesor yang tersedia (maksimum waktu berjalan semua prosesor) atau akumulasi waktu eksekusi dari semua prosesor (penjumlahan waktu berjalan semua prosesor). Karena tugas-tugas yang ditugaskan ke sebuah prosesor harus dieksekusi dalam kecepatan yang sama, masalah optimal selanjutnya menjadi cukup sederhana. Optimal dapat dicapai dengan memilih prosesor yang paling efisien dan menetapkan semua tugas padanya. Kami memilih untuk mengoptimalkan masalah sulit mengoptimalkan waktu berjalan bersamaan (lihat Persamaan (5.1)).

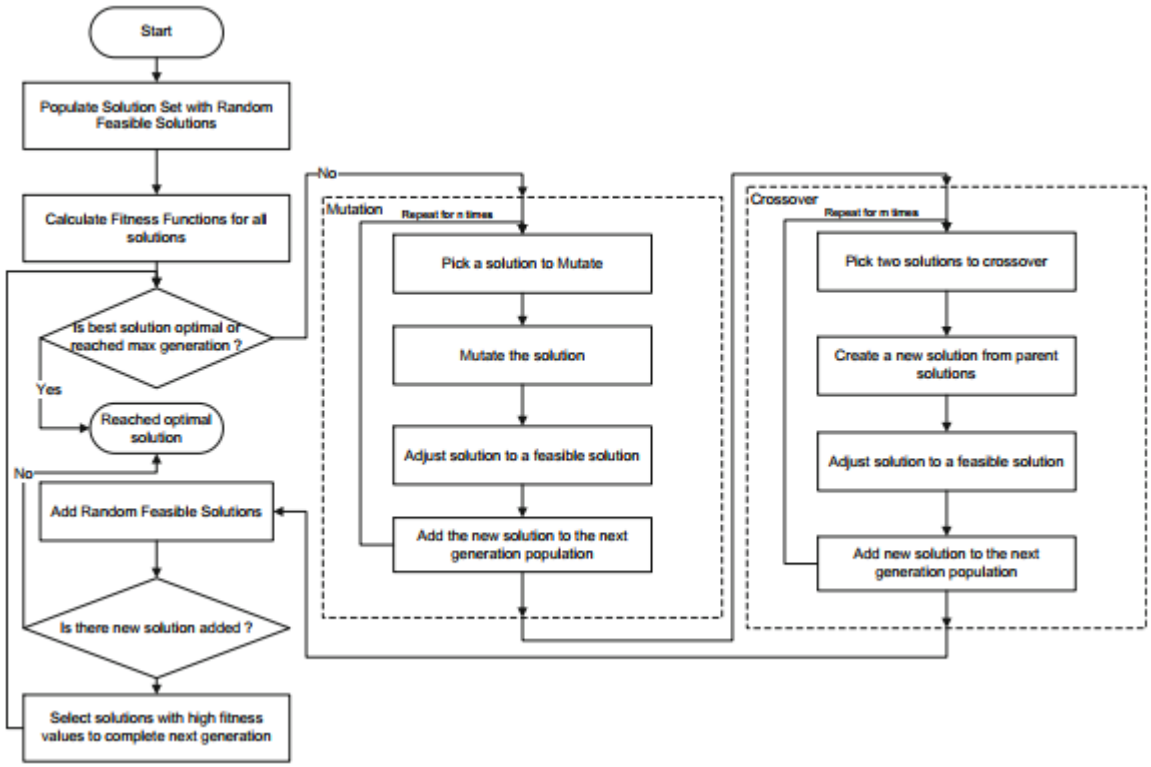
Selain fungsi minimal, standar deviasi pada waktu eksekusi juga dapat digunakan sebagai fungsi tujuan untuk dioptimalkan. Ini mengukur jarak dari waktu eksekusi masing-masing prosesor ke rata-rata. Idenya adalah untuk membuat semua prosesor berbagi beban kerja dan menerapkan waktu eksekusi mereka mendekati media. Ini adalah fungsi tujuan yang baik secara umum tetapi mungkin tidak berfungsi di lingkungan prosesor yang heterogen. Dalam pengaturan yang sangat beragam, efisiensi energi dan eksekusi prosesor dapat berbeda secara signifikan satu sama lain. Mungkin ada solusi optimal bahwa tidak ada pekerjaan yang diberikan ke prosesor yang kurang efisien. Fungsi minimal yang sederhana efisien dalam penghitungan dan fleksibel untuk mencakup sebagian besar skenario.

5.3 ALGORITMA GENETIKA UNTUK PENJADWALAN TUGAS HIJAU

Algoritma Genetika (GA) adalah algoritma pencarian multi-solusi berbasis hadiah. Ini adalah cabang dari algoritma evolusi yang terinspirasi bio (EA). Ini telah berhasil diterapkan ke banyak

aplikasi. Dibandingkan dengan algoritme pencarian solusi tunggal lainnya seperti Linear Programming LP, k-opt, ada beberapa solusi layak yang secara bersamaan berkembang menuju solusi terbaik dalam proses pencarian GA.

Alih-alih dari satu titik pencarian, GA secara acak memilih beberapa titik awal di ruang pencarian. Solusi awal ini diatur ke dalam populasi. Kemudian, proses pencarian GA mengembangkan semua solusi dalam populasi secara bertahap menuju solusi optimal akhir. Evolusi diatur dalam beberapa generasi. Untuk mengevolusikan generasi, proses pencarian GA (lihat Gambar 5.1) menerapkan operasi crossover, mutasi, random reseeding, dan elite filtering ke solusi dalam populasi. Operator crossover meniru orang tua yang menghasilkan anak secara alami. Berdasarkan domain masalah, metode pemuliaan digunakan untuk membuat solusi "anak" dari dua solusi induk. Solusi anak mewarisi atribut tertentu dari kedua orang tua. Tujuan umumnya adalah menciptakan solusi anak yang memiliki sifat baik dari kedua orang tua dan lebih baik dari kedua orang tua.



Gambar 5.1. Algoritma genetika

Untuk menghasilkan solusi baru, operator mutasi unary memodifikasi keadaan (-s) dari satu atau sejumlah kecil komponen dari solusi yang ada. Seringkali, solusi yang baru dihasilkan jauh berbeda dari solusi asli dan bahkan mungkin bukan solusi yang valid. Itu bisa membawa pencarian ke daerah yang belum pernah dikunjungi sebelumnya. Ini dapat digunakan untuk memecahkan perangkap optimal lokal dan memindahkan solusi dengan cepat. Selain operator mutasi dan persilangan, solusi baru juga dihasilkan secara acak dalam proses evolusi secara umum. Ini untuk lebih memperluas ruang pencarian dan berfungsi sebagai jaminan tambahan dari pencarian global.

Pemfilteran elit adalah mekanisme penghargaan dalam pencarian GA. Ini memastikan solusi yang lebih baik bertahan dari proses evolusi dan berpartisipasi dalam evolusi generasi berikutnya. Ini memandu pencarian menuju solusi optimal atau mendekati optimal.

Karena GA memberlakukan sangat sedikit persyaratan untuk masalah pengoptimalan, GA telah digunakan secara luas di banyak bidang dan mencapai hasil yang sangat baik. Namun kualitas solusi dan kecepatan pencariannya tidak selalu memuaskan. Banyak perbaikan telah dilakukan untuk meningkatkan performa GA, seperti menambahkan pencarian lokal, menambahkan kemampuan adaptif, dan hibridisasi dengan algoritma lain. Kami memperkenalkan peningkatan lain pada pencarian GA yang dapat mencapai hasil yang jauh lebih baik.

Algoritma Genetika Peningkatan Harga Bayangan

Keacakan adalah kunci untuk menjadikan GA algoritma pencarian global yang sukses. Tetapi keacakan juga memperkenalkan banyak perhitungan yang tidak perlu dalam proses pencarian. Operasi GA memperkenalkan banyak solusi acak yang lebih buruk. Pemborosan daya komputasi ini menyebabkan kecepatan pencarian yang lambat dan hasil berkualitas rendah. Kami menggunakan harga bayangan untuk meningkatkan efisiensi dan tetap mempertahankan keacakan yang diperlukan untuk pencarian GA. Harga bayangan diperkenalkan pada algoritma pencarian Linear Programming (LP) pada awalnya. Dalam sebuah solusi, setiap komponen yang membentuk solusi tersebut memiliki harga yang terkait dengannya. Jumlah semua harga komponen adalah biaya solusi. Ini adalah nilai nyata yang konkret. Dalam proses pencarian LP, harga bayangan digunakan sebagai ukuran potensi kontribusi komponen terhadap solusi. Ini adalah nilai yang dibuat-buat yang hanya berguna dalam proses pencarian, sehingga disebut harga bayangan. Harga bayangan adalah kontribusi terhadap fungsi tujuan yang dapat dilakukan dengan melonggarkan kendala sebesar satu unit. Kendala yang berbeda memiliki harga bayangan yang berbeda, dan setiap kendala memiliki harga bayangan. Setiap harga bayangan kendala berubah seiring dengan pencarian solusi optimal. Dalam proses pencarian GA, nilai fitness digunakan untuk mengukur kualitas suatu solusi.

Ini digunakan untuk membandingkan solusi dan memutuskan solusi mana yang cocok untuk bertahan dalam proses evolusi (hadiah). Bahkan, itu adalah satu-satunya pengukuran yang digunakan dalam proses pencarian. Operasi evolusi sepenuhnya didasarkan pada keacakan. Solusi yang berpartisipasi dipilih secara acak, komponen dipilih secara acak, dan parameter operasi dipilih secara acak. Tidak ada arahan umum untuk mengembangkan solusi dan solusi yang baru dihasilkan sangat beragam. Ini tidak terlalu efisien.

Untuk mengatasi kekurangan ini, kami memperkenalkan konsep harga bayangan ke dalam proses pencarian GA. Kami mendefinisikan harga bayangan sebagai peningkatan potensial relatif terhadap nilai kesesuaian solusi dengan perubahan komponen. Harga bayangan ditentukan untuk suatu komponen. Harga bayangan solusi adalah jumlah dari semua harga bayangan komponen. Harga bayangan yang kami definisikan di sini adalah nilai semu yang diambil dari status komponen saat ini. Hal ini tidak secara langsung terkait dengan biaya yang sebenarnya tetapi peningkatan potensial dari perubahan. Kontribusi mungkin atau mungkin tidak dapat direalisasikan karena solusi harus divalidasi setelah perubahan

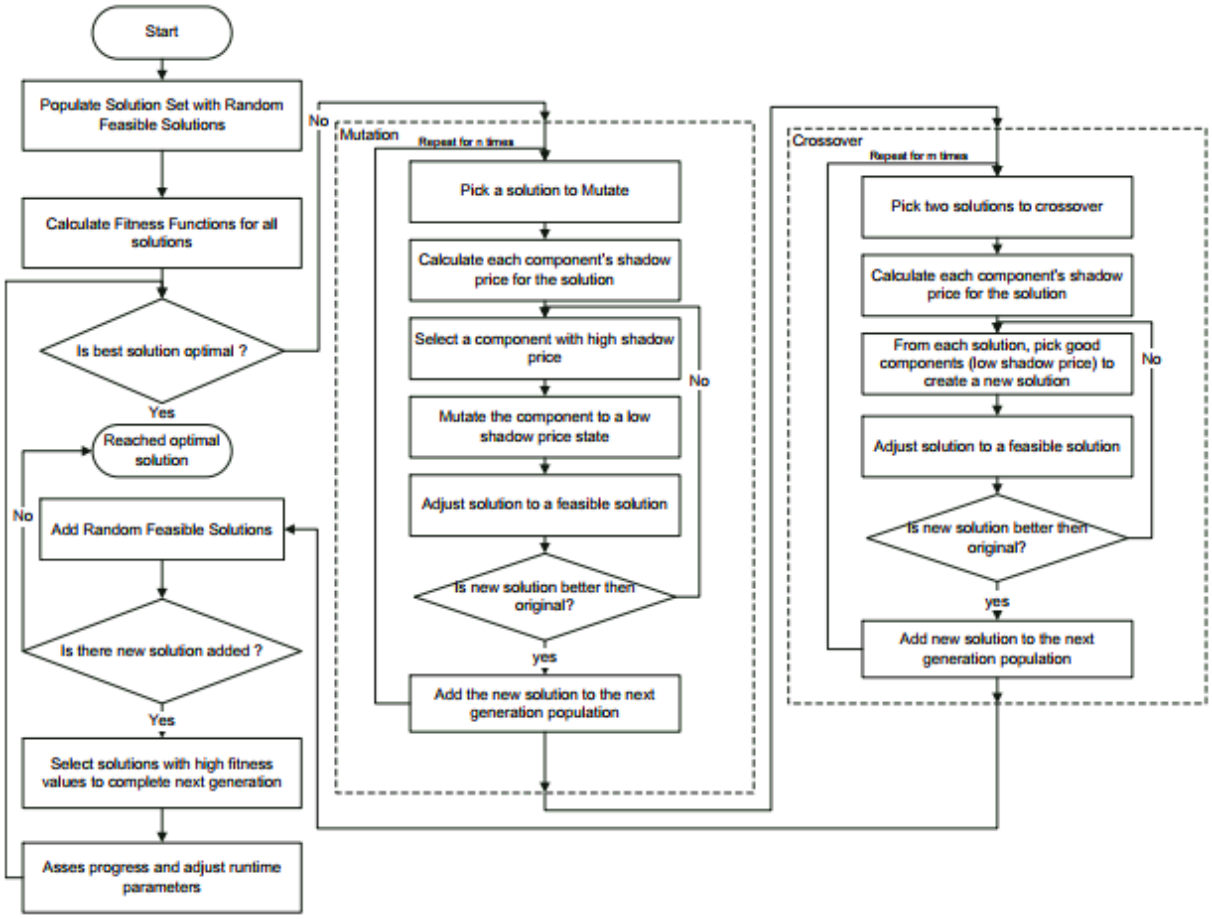
komponen. Perubahan tersebut dapat berupa perubahan nilai, perubahan posisi, atau perubahan aplikatif lainnya. Komponen dengan harga bayangan yang tinggi berpotensi memberikan kontribusi yang besar terhadap nilai fitness. Kami menggunakan harga bayangan untuk membandingkan komponen secara langsung, hubungan mereka, dan kontribusi terhadap solusi yang lebih baik.

Berdasarkan masalah yang berbeda, harga bayangan dapat memiliki arti atau nilai yang berbeda. Dalam masalah travelling salesman, ini dapat berupa kemungkinan pengurangan jarak dari perubahan kota kunjungan berikutnya dari kota saat ini. Dalam manufaktur, harga bayangan dapat berupa biaya material, waktu, dll.

Dengan diperkenalkannya harga bayangan, kami dapat meningkatkan kinerja operator GA secara signifikan. Operasi crossover bermaksud untuk melewatkan komponen yang baik dari orang tua ke anak sehingga solusi anak lebih baik dari orang tua. Meneruskan komponen yang dipilih secara acak ke anak-anak dapat menghasilkan banyak solusi anak yang lebih buruk dan menghabiskan banyak sumber daya komputasi. Untuk memperbaikinya, kita dapat menggunakan harga bayangan untuk mengevaluasi dan membandingkan komponen. Karena harga bayangan mengkuantifikasi kontribusi komponen, kita dapat secara selektif memberikan komponen bagus yang memberikan kontribusi lebih besar kepada anak. Hal ini dapat meningkatkan kemungkinan menghasilkan anak yang lebih baik secara signifikan. Untuk menghindari jebakan optimal lokal, perlu untuk membangun beberapa keacakan ke dalam proses pemilihan komponen harga bayangan tinggi.

Mutasi mengembangkan solusi dengan mengubah status satu atau lebih komponen. Ini adalah operasi yang sangat efektif yang dapat mengubah solusi secara dramatis. Sering kali, hanya mutasi yang digunakan di GA. Ini kontraproduktif dan pemborosan sumber daya komputasi untuk mengubah komponen dari keadaan baik menjadi lebih buruk. Memilih komponen dan arah untuk berkembang adalah kunci efisiensi. Karena harga bayangan menentukan kontribusi potensial komponen dan memungkinkan perbandingan komponen, kita dapat memilih komponen berpotensi tinggi untuk bermutasi dan memutasikannya ke status berpotensi rendah. Alih-alih evolusi acak pada komponen acak, mutasi yang mengaktifkan harga bayangan dapat menghasilkan banyak solusi yang lebih baik dan meningkatkan efisiensi sumber daya komputasi. Untuk menghindari jebakan optimal lokal, keacakan perlu dibangun ke dalam pemilihan komponen dan pengaturan arah.

Dalam algoritma genetik ditingkatkan harga bayangan baru (S PGA), kami menetapkan sistem dua pengukuran: nilai kebugaran digunakan untuk mengevaluasi kebaikan solusi secara keseluruhan, dan harga bayangan digunakan untuk mengevaluasi kebaikan komponen. Kami meningkatkan operasi GA dengan harga bayangan. Kami menggunakan harga bayangan komponen untuk memilih komponen yang akan dioperasikan dan menetapkan arah untuk berkembang. Kami mengarahkan pencarian menuju solusi optimal dan tetap mempertahankan keacakan yang diperlukan dalam proses pencarian. S PGA (lihat Gambar 5.2) jauh lebih efisien dalam konvergen ke solusi optimal dan dapat memberikan hasil yang lebih baik.



Gambar 5.2. Algoritma Genetika yang Ditingkatkan Harga Bayangan

Penjadwalan Tugas Ramah Lingkungan Menggunakan S PGA

Tujuan utama penjadwalan tugas hijau kami (lihat Persamaan (5.1)) adalah untuk menjadwalkan *m* tugas pada *n* komputer sedemikian rupa sehingga waktu eksekusi bersamaan minimal dan total konsumsi energi kurang dari atau sama dengan *E.*

Terbukti bahwa paling efisien, baik untuk minimisasi energi maupun minimisasi waktu eksekusi, untuk menjadwalkan tugas pada prosesor yang sama dengan kecepatan yang sama. Masalah optimisasi penjadwalan tugas hijau dipecah menjadi dua sub masalah, mendistribusikan kendala energi optimal *E* ke *n* komputer dan menetapkan *m* tugas optimal ke *n* komputer. Masalah sub optimisasi ketiga yang asli, meminimalkan waktu eksekusi untuk prosesor *i* dengan tugas *mi* dan batas energi *Ei,* dapat diselesaikan secara langsung menggunakan Persamaan. (5.7) dan kecepatan dapat dihitung menggunakan Persamaan. (5.10).

Pengkodean sangat penting dalam proses pencarian GA karena berdampak langsung pada kinerja. Skema dalam masalah ini sangat mudah. Karena urutan prosesor dalam solusi tidak berdampak pada solusi, daftar prosesor dan tugas yang diberikan padanya dapat digunakan untuk mewakili solusi. Definisi prosesor mencakup atributnya, seperti *Ci, αi,* kecepatan minimal, kecepatan maksimal, dll. Tugas yang terkait dengan prosesor juga dapat disimpan dalam daftar karena urutannya juga tidak penting.

Ada dua langkah untuk membangun solusi, mendistribusikan kendala energi dan menetapkan tugas. Itu tidak memengaruhi solusi tugas mana yang diselesaikan terlebih dahulu. Namun kedua tugas tersebut harus diselesaikan sebelum nilai fitness dapat dihitung untuk sebuah solusi.

Untuk mengatasi masalah penjadwalan, kita dapat memperlakukannya sebagai dua masalah pengoptimalan bersarang atau masalah pengoptimalan dengan dua subtugas. Dalam skenario masalah optimisasi bersarang, satu sub masalah akan dipilih sebagai masalah induk dan digunakan untuk mendorong masalah anak lainnya. Misalnya, jika kita memilih distribusi kendala energi sebagai masalah induk, proses pencarian dimulai dengan membuat berbagai kombinasi penugasan energi ke prosesor. Setiap penetapan kendala energi akan diperlakukan sebagai masalah optimisasi yang terpisah dan diselesaikan secara individual. Berbagai penugasan tugas dievaluasi dan penugasan dengan waktu eksekusi bersamaan paling sedikit adalah nilai kebugaran untuk penugasan energi. Proses pencarian mengembangkan penugasan energi induk dan mencari kombinasi tugas terbaik untuk setiap penugasan baru. Proses berulang sampai solusi optimal ditemukan.

Kami juga dapat memperlakukan dua tugas pengoptimalan sebagai dua parameter terpisah dalam masalah pengoptimalan yang sama dan membuat model datar. Dalam model bersarang, induk mencari penetapan energi yang optimal dan anak mencari penetapan tugas terbaik dalam penetapan energi induk. Dalam model datar, kedua parameter bekerja sama untuk mengoptimalkan tujuan yang sama yaitu meminimalkan waktu eksekusi solusi untuk semua prosesor. Dengan demikian, kita dapat mengabaikan hubungan antara kedua parameter ini dan hanya berfokus pada hubungan dari kedua parameter tersebut ke solusinya. Kami dapat menyetel satu parameter pada satu waktu dan memutar. Proses dapat diulang sampai solusi optimal ditemukan.

Masalah optimisasi bersarang sulit dipecahkan dan membutuhkan lebih banyak waktu untuk berkumpul. Sebagai perbandingan, model datar lebih mudah diselesaikan karena hanya ada satu fungsi objektif. Kompleksitasnya adalah bahwa ada lebih banyak parameter dalam operasi GA. Mengoptimalkan model bersarang menggunakan pencarian pohon dan mengoptimalkan model datar menggunakan pencarian linier dengan parameter berputar.

Untuk bekerja dengan model datar, kami mendefinisikan dua operasi mutasi, mutasi energi dan mutasi tugas. Ada dua mutasi sub energi, pertukaran energi antara dua prosesor dan memindahkan sebagian energi dari satu prosesor ke prosesor lainnya. Ada juga dua mutasi sub tugas, menukar sepasang tugas antara dua prosesor dan memindahkan satu tugas dari satu prosesor ke prosesor lainnya. Prosesor dan tugas dipilih secara acak dalam operasi. Tidak ada preferensi atau arahan untuk memindahkan proses pencarian.

Operasi mutasi kami yang ditingkatkan hanya memindahkan sejumlah energi dari satu prosesor ke prosesor lainnya. Karena tujuannya adalah untuk meminimalkan waktu eksekusi bersamaan dan lebih banyak energi dapat meningkatkan kecepatan, kami ingin memindahkan sebagian energi dari prosesor waktu kerja pendek ke prosesor waktu kerja panjang. Prosesor waktu jangka panjang akan mendapat manfaat dari energi tambahan dan mempersingkat waktu pengoperasian. Tetapi prosesor waktu kerja pendek mungkin tidak memiliki energi ekstra untuk diberikan. Mungkin ada beberapa alasan yang menyebabkan prosesor

menggunakan lebih sedikit waktu, seperti prosesor sangat efisien dan dapat berjalan sangat cepat dengan sedikit energi, prosesor diberi energi dalam jumlah besar, atau prosesor diberi tugas kecil. Jadi run time yang singkat tidak dapat digunakan untuk memilih prosesor donor energi. Kombinasi energi yang lebih tinggi dan waktu tempuh yang lebih sedikit membuat kriteria pemilihan yang baik.

Dalam definisi kami, harga bayangan mewakili potensi komponen. Di sini, harga bayangan adalah kombinasi dari waktu kerja prosesor dan energi yang diberikan. Waktu berjalan lebih diutamakan daripada energi karena kami memilih prosesor donor energi. Harga bayangan prosesor tinggi ketika memiliki waktu kerja yang singkat dan energi yang besar. Harga bayangan prosesor rendah ketika memiliki waktu pengoperasian yang lama atau waktu pengoperasian yang singkat dan energi yang lebih kecil. Kami ingin mengubah prosesor dari status harga bayangan tinggi ke status harga bayangan rendah. Arah mutasi yang ditentukan oleh harga bayangan adalah untuk memutasi prosesor dengan waktu kerja di bawah rata-rata ke waktu kerja yang lebih lama atau keadaan energi yang lebih sedikit.

Kami sebelumnya menggunakan konsumsi energi rata-rata per instruksi sebagai harga bayangan untuk setiap prosesor dan berhasil memecahkan konsumsi energi minimal dengan kendala waktu masalah penjadwalan hijau. Definisi ini, konsumsi energi rata-rata per instruksi atau waktu rata-rata yang dihabiskan per instruksi, tidak berlaku untuk mutasi tugas di sini karena setiap prosesor dapat ditugaskan dengan energi dan tugas yang berbeda. Energi rata-rata atau waktu per instruksi tidak dapat digunakan untuk membandingkan antar prosesor. Konsumsi energi rata-rata yang tinggi per instruksi dapat terjadi untuk prosesor dengan berbagai penetapan energi atau tugas. Fakta yang sama berlaku untuk waktu yang dihabiskan per instruksi.

Tujuan mutasi tugas adalah memindahkan tugas dari prosesor yang berjalan lama ke prosesor yang berjalan singkat. Pemroses donor tugas mudah dipilih. Ini bisa menjadi salah satu prosesor jangka panjang. Prosesor penerima harus menjadi salah satu prosesor waktu berjalan singkat. Kita harus sangat berhati-hati dalam memilih prosesor penerima karena dapat meningkatkan waktu kerjanya secara dramatis. Karena kita tidak mengatur ulang energi dalam mutasi tugas ini, prosesor penerima yang ideal adalah prosesor yang energi atau waktu eksekusinya tidak terlalu sensitif terhadap peningkatan tugas. Artinya, peningkatan tugas bukanlah faktor yang paling berpengaruh dalam perhitungan waktu atau energi eksekusi prosesor. Rumus (5.7) menunjukkan perhitungan waktu eksekusi dengan energi tetap dan Persamaan. (5.9) menunjukkan perhitungan energi dengan kecepatan yang diketahui. Keduanya adalah fungsi eksponensial. Dalam fungsi eksponensial, eksponen memiliki pengaruh yang jauh lebih besar terhadap hasil daripada basis. Dalam kedua Persamaan. (5.7) dan Persamaan. (5.9), hitungan instruksi tugas ada di basis dan α ada di eksponen. Karena α adalah angka positif dan lebih besar atau sama dengan 3, hal itu dapat menghasilkan dampak yang lebih besar terhadap waktu eksekusi dan konsumsi energi. Saat membandingkan 2 prosesor dengan tugas yang sama, prosesor dengan α lebih besar mengonsumsi lebih banyak energi jika kecepatannya sama atau membutuhkan waktu eksekusi lebih lama jika energinya sama. Jadi, lebih disukai untuk menambahkan tugas ke prosesor dengan α yang lebih kecil karena dapat menyebabkan peningkatan waktu eksekusi yang jauh lebih kecil. Kami

mendefinisikan harga bayangan sebagai kombinasi dari waktu eksekusi dan α. Kami ingin mengubah tugas dari prosesor dengan waktu eksekusi yang lama menjadi prosesor dengan waktu eksekusi yang singkat dan α yang lebih kecil.

Algoritma peningkatan harga bayangan kami mengikuti kerangka kerja algoritma GA standar. Untuk menghindari jebakan optimal lokal, kami menggabungkan mutasi yang ditingkatkan dengan operasi mutasi standar.

Algoritma 2. Bayangan harga GA
1: Validasi setidaknya ada satu solusi yang layak;
2: Membangun populasi awal;
3: selama kriteria berhenti belum terpenuhi lakukan
4: Pilih subpopulasi untuk menerapkan salah satu operasi berikut secara acak;
5: Operasi mutasi perpindahan energi;
6: Pilih dua prosesor secara acak;
7: Pindahkan sebagian energi dari satu prosesor ke prosesor lainnya;
8: Validasi solusi baru;
9: Operasi mutasi pertukaran energi;
10: Pilih dua prosesor secara acak;
11: Pertukaran penugasan energi di antara mereka;
12: Validasi solusi baru;
13: Operasi mutasi perpindahan tugas;
14: Pilih dua prosesor secara acak;
15: Pilih tugas secara acak dari satu prosesor;
16: Pindahkan tugas yang dipilih secara acak dari satu prosesor ke prosesor lainnya;
17: Validasi solusi baru;
18: Operasi mutasi pertukaran tugas;
19: Pilih dua prosesor secara acak;
20: Pilih tugas secara acak dari setiap prosesor;
21: Pertukaran tugas yang dipilih antara dua prosesor;
22: Validasi solusi baru;
23: Operasi mutasi energi peningkatan harga bayangan:
24: Urutkan semua prosesor berdasarkan waktu eksekusi;
25: Membagi prosesor menjadi 2 set, prosesor jangka panjang dan prosesor jangka pendek;
26: Buat subset dari prosesor waktu jangka panjang untuk membuat kumpulan prosesor penerima energi S r;
27: Pilih secara acak satu prosesor dari S r sebagai prosesor penerima Pr;
28: Persingkat ulang set prosesor waktu kerja pendek berdasarkan penetapan energi;
29: Buat subset dari prosesor waktu kerja singkat untuk membuat kumpulan prosesor donor energi tinggi S d;
30: Pilih secara acak satu prosesor dari S d sebagai prosesor yang menyumbangkan energi Pd;
31: Pindahkan sebagian energi dari Pd ke Pr;
32: Validasi solusi baru;

33: Operasi mutasi tugas peningkatan harga bayangan:
34: Urutkan semua prosesor berdasarkan waktu eksekusi;
35: Membagi prosesor menjadi 2 set, prosesor jangka panjang dan prosesor jangka pendek;
36: Buat subset dari prosesor jangka panjang untuk membuat tugas yang menyumbangkan kumpulan prosesor S d;
37: Pilih secara acak satu prosesor dari S d sebagai prosesor donor Pd;
38: Persingkat kembali set prosesor waktu kerja pendek berdasarkan nilai α prosesor;
39: Buat subset dari prosesor waktu berjalan singkat untuk menetapkan tugas nilai α kecil yang menerima kumpulan prosesor S r;
40: Pilih satu prosesor secara acak dari S r sebagai prosesor penerima tugas Pd;
41: Pilih satu tugas secara acak dari Pd dan pindah ke Pr;
42: Validasi solusi baru;
43: Tambahkan solusi acak;
44: Saring dan bangun generasi berikutnya;
45: akhir sementara
46: kembali Solusi(-s)

Harga bayangan mewakili keadaan komponen relatif terhadap solusi saat ini. Ini mengukur potensi kontribusi komponen. Ini digunakan untuk membandingkan komponen dalam solusi dalam proses pencarian. Itu bisa mengambil banyak bentuk yang berbeda. Ini bisa menjadi pemborosan material yang sebenarnya dalam masalah pemotongan stok, angka relatif terhadap solusi dalam masalah penjual keliling, atau fungsi dalam masalah pengurangan stok.

Dalam masalah penjadwalan hijau ini, harga bayangan tertanam dalam operasi mutasi karena kerumitannya. Ini adalah prosedur. Ini mengukur waktu eksekusi prosesor, konsumsi energi, dan atribut prosesor α. Ini memungkinkan pemilihan komponen potensial besar untuk bermutasi ke keadaan potensial rendah. Harga bayangan dapat menghilangkan banyak perhitungan yang tidak perlu. Ini dapat sangat meningkatkan kecepatan pencarian dan kualitas solusi. Dalam GA harga bayangan kami ditingkatkan (lihat Algoritma 2), kami masih menggunakan kesesuaian untuk memfilter solusi dengan menggunakan perintah yang ditentukan pada Baris 44. Kami menerapkan sistem dua pengukuran, kesesuaian untuk mengukur solusi keseluruhan dan harga bayangan untuk mengukur komponen, untuk meningkatkan kecepatan dan hasil GA.

5.4 ANALISIS KINERJA

Kami melakukan studi komparatif yang komprehensif antara GA dan harga bayangan baru kami yang ditingkatkan GA. Kedua algoritme mengikuti kerangka GA standar dan identik kecuali operasi mutasi. S PGA dikodekan berdasarkan Algoritma 2 dan mengimplementasikan keenam operasi mutasi. Hanya empat operasi mutasi pertama yang digunakan dalam GA standar.

Dengan menggunakan perintah di Baris 4 dari Aloritma 2 kami memeriksa keberadaan solusi yang valid. Rumus (5.5) dan (5.9) menunjukkan bahwa konsumsi energi berada pada level terendah ketika kecepatan diminimalkan untuk prosesor tertentu. Untuk memeriksa

apakah sebuah prosesor dapat menyelesaikan tugas dengan energi terbatas, kita hanya perlu mengujinya pada kecepatan terendahnya. Untuk memeriksa keberadaan setidaknya satu solusi yang valid, kami menguji semua tugas untuk setiap prosesor pada kecepatan terendahnya dan membandingkan konsumsi energi. Jika ada satu prosesor mengkonsumsi kurang dari atau sama dengan kendala energi, setidaknya ada satu solusi yang valid untuk masalah tersebut. Jika tidak, tidak ada solusi yang layak untuk masalah tersebut dan algoritme dibatalkan.

Kami membuat kode dan menguji kedua algoritme di Microsoft C#. Semua percobaan dijalankan pada laptop Lenovo Thinkpad T410 yang dilengkapi dengan CPU Intel Core i5-M520 2,4 GHz dan memori 4 GB yang menjalankan Windows 7.

Untuk membuat kasus uji, kita memerlukan definisi dan tugas prosesor. Kami memilih 20 CPU komersial terbaru yang dirilis untuk eksperimen kami (lihat Tabel 5.1). Kami menggunakan kecepatan yang dipublikasikan sebagai kecepatan minimum prosesor. Angka acak antara 5% dan 25% digunakan sebagai peningkatan kecepatan untuk menentukan kecepatan maksimum prosesor. Konstanta C dihasilkan secara acak antara 1 dan 100. Karena α berdampak besar pada konsumsi energi CPU, maka dipilih secara acak dalam rentang sempit antara 3 dan 5. Ini untuk menghindari perbedaan besar di antara prosesor. Jumlah instruksi untuk tugas juga dihasilkan secara acak antara 500 dan 100.000 instruksi. Untuk meningkatkan kualitas, kami menggunakan layanan penghasil nomor acak yang tersedia untuk umum alih-alih pustaka C#.

Tabel 5.1. Prosesor

ID	Processor	Year	Min Speed (MIPS)	Speed up (%)		Max Speed (MIPS)		*C*		*α*
1	DEC Alpha 21064 EV4	1992	300	9		327		84		4.08
2	Intel Pentium III	1999	1354	15		1557		7		3.67
3	AMD Athlon	2000	3561	23		4380		8		4.51
4	AMD Athlon XP 2400+	2002	5935	14		6766		86		3.94
5	Pentium 4 Ex.Ed.	2003	9726	7	10407		74		3.50	
6	AMD Athlon FX-57	2005	12000	9	13080		50		3.41	
7	AMD Athlon 64 3800+ X2 (Dual Core)	2005	14564	13		16457		60		3.74
8	ARM Cortex A8	2005	2000	18		2360		52		4.57
9	Xbox360 IBM ”Xenon” (Triple Core)	2005	19200	20		23040		55		4.92
10	AMD Athlon FX-60 (Dual Core)	2006	18938	17		22157		62		4.17
11	Intel Core 2	2006	27079	21		32766		19		4.03

	(Extreme X6800)									
12	Intel Core 2 (Extreme QX6700)	2006	49161	16		57027		22		4.33
13	PS3 Cell BE ((PPE only))	2006	10240	23		12595		45		3.13
14	P.A. Semi (PA6T-1682M)	2007	8800	25		11000		11		3.67
15	Intel Core 2 (Extreme QX9770)	2008	59455	17		69562		92		4.64
16	Intel Core i7 (Extreme 965EE)	2008	76383	18		90132		11		4.08
17	AMD Phenom II (X4 940 Black Ed.)	2009	42820	15		49243		42		4.57
18	AMD Phenom II (X6 1090T)	2010	68200	12		76384		77		3.20
19	Intel Core i7 (Extreme Ed. i980EE)	2010	147600	14		168264		29		3.82
20	IBM 5.2-GHz z196	2010	52286	15		60129		66		3.90

Kasus percobaan dibuat menggunakan berbagai kombinasi prosesor dan tugas. Kendala energi untuk setiap kasus eksperimen dibuat secara acak, divalidasi, dan dipersingkat untuk memastikan bahwa hanya sedikit prosesor yang menganggur dalam solusi optimal.

Kami mempelajari kinerja algoritma dalam dua aspek, kualitas hasil dan kecepatan konvergensi. Untuk kualitas hasil, kami menguji algoritma dengan berbagai kasus uji di bawah batasan energi tetap dan pembangkitan evolusi tetap.

Tabel 5.2–5.6 menunjukkan hasil uji perbandingan antara *GA* dan *S PGA*. Agar mudah dibaca, hanya bagian bilangan bulat dari data yang ditampilkan. Hitungan prosesor berkisar dari 10 hingga 50. Hitungan tugas *R* berkisar dari 500 hingga 5000. Batas generasi *maks (Gmax)* adalah 500, 1000, 1500, 2000, 3000, dan 5000. Semua kombinasi hitungan tugas *R* dan generasi batas *Gmax* diuji untuk setiap pengaturan jumlah prosesor.

Setiap test case dijalankan setidaknya 10 kali. Hasil dirata-rata dan dilaporkan. Persentase peningkatan dari solusi optimal *S PGA (Tspga)* dibandingkan solusi optimal *GA (Tga)* dilaporkan dalam tabel. Karena tujuannya adalah untuk meminimalkan waktu eksekusi bersamaan, angka positif menunjukkan solusi terbaik *GA* membutuhkan waktu lama daripada *S PGA* dan hasil *S PGA* lebih baik daripada *GA*.

Tabel 5.2–5.6 menunjukkan semua hasil positif. Solusi terbaik *S PGA* menggunakan waktu eksekusi yang lebih sedikit daripada solusi terbaik *GA* di semua kasus uji. *S PGA* mencapai solusi yang lebih baik daripada *GA*.

Tabel 5.2. Peningkatan Waktu *S PGA* atas $GA(Tga - Tspga) \times \frac{100}{Tga}$ untuk 10 Prosesor

G_{max}	*R* = 500	*R* = 1000	*R* = 1500	*R* = 2000	*R* = 3000	*R* = 5000
500	44	25	15	8	5	1
1000	85	56	30	22	12	2
1500	84	76	51	37	19	8
2000	74	78	71	50	27	13
3000	57	70	77	75	45	24
5000	23	58	68	72	76	50

Tabel 5.3. Peningkatan Waktu *S PGA* atas $GA(Tga - Tspga) \times \frac{100}{Tga}$ untuk 20 Prosesor

G_{max}	*R* = 500	*R* = 1000	*R* = 1500	*R* = 2000	*R* = 3000	*R* = 5000
500	76	58	49	39	28	26
1000	69	79	70	60	55	41
1500	57	79	79	74	62	49
2000	49	78	81	80	73	56
3000	42	70	76	80	79	67
5000	31	56	69	75	78	77

Untuk mempelajari kecepatan konvergensi algoritme, kami memutar ulang semua kasus uji dengan batasan energi yang sama. Alih-alih membatasi generasi maksimal, kami menetapkan nilai kebugaran target untuk algoritme. Pencarian hanya berhenti ketika waktu eksekusi solusi terbaik memenuhi nilai target. Algoritme dapat memakan waktu atau generasi sebanyak yang diperlukan untuk mencapai target. Waktu eksekusi rata-rata dari kasus uji di atas digunakan sebagai nilai target.

Tabel 5.4. Peningkatan Waktu *S PGA* atas $GA(Tga - Tspga) \times \frac{100}{Tga}$ untuk 30 Prosesor

G_{max}	*R* = 500	*R* = 1000	*R* = 1500	*R* = 2000	*R* = 3000	*R* = 5000
500	88	75	61	49	25	18
1000	90	86	79	72	54	36
1500	85	90	86	80	68	53
2000	81	87	90	86	82	68
3000	69	82	88	89	88	79
5000	46	76	82	86	90	86

Tabel 5.5. Peningkatan Waktu *S PGA* atas $GA(Tga - Tspga) \times \frac{100}{Tga}$ untuk 40 Prosesor

G_{max}	*R* = 500	*R* = 1000	*R* = 1500	*R* = 2000	*R* = 3000	*R* = 5000
500	81	67	56	45	31	39
1000	81	83	74	68	60	53
1500	76	85	84	79	73	61

2000	70	83	87	85	80	66
3000	58	79	84	86	86	74
5000	43	71	79	82	85	85

Tabel 5.6. Peningkatan Waktu *S PGA* atas $GA(Tga - Tspga) \times \frac{100}{Tga}$ untuk 50 Prosesor

G_{max}	*R* = 500	*R* = 1000	*R* = 1500	*R* = 2000	*R* = 3000	*R* = 5000
500	88	74	69	63	52	37
1000	90	85	81	76	72	61
1500	88	91	89	84	80	72
2000	85	89	91	88	84	76
3000	80	85	89	91	90	83
5000	61	80	85	88	91	89

Tes dijalankan untuk setiap kombinasi hitungan prosesor *(Pc)* dan hitungan tugas *R*. Setiap tes dijalankan setidaknya sepuluh kali. Hasil dirata-rata dan dilaporkan. Tabel 5.7 menunjukkan penghematan waktu pencarian *(STspga) S PGA* dibandingkan waktu pencarian *GA (S Tga), (S Tga – STspga*). Nilai positif menunjukkan *GA* membutuhkan waktu lebih lama dari S PGA untuk mencapai hasil yang setara. *S PGA* lebih cepat dengan nilai positif. Tabel 5.8 membandingkan generasi evolusi yang digunakan dari *GA (Gga)* atas *S PGA (Gspga), (Gga Gspga)*. Nilai positif menyatakan bahwa GA membutuhkan lebih banyak generasi evolusi daripada *S PGA* untuk mencapai solusi yang ditargetkan. Kedua tabel menunjukkan S PGA lebih cepat dari GA. Tabel 5.7 mengukur kecepatan waktu pencarian dan Tabel 5.8 mengukur kecepatan evolusi generasi.

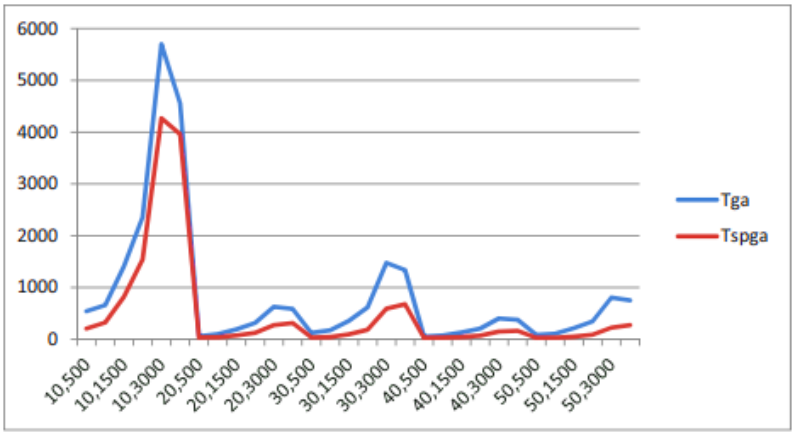
Tabel 5.7. *S PGA* Peningkatan Kecepatan Pencarian dalam Waktu, $ST_{ga} - ST_{spga}$

Pc	*R* = 500	*R* = 1000	*R* = 1500	*R* = 2000	*R* = 3000	*R* = 5000
10	3	3	10	18	33	60
20	3	8	12	15	36	52
30	5	7	10	13	21	42
40	6	8	12	17	31	43
50	8	10	12	16	20	38

Tabel 5.8. Peningkatan Kecepatan Pencarian PGA S dalam Generasi, $G_{ga} - G_{spga}$

Pc	*R* = 500	*R* = 1000	*R* = 1500	*R* = 2000	*R* = 3000	*R* = 5000
10	791	538	783	845	782	407
20	945	1471	1424	1157	1456	1012
30	1154	1172	1273	1145	1115	1132
40	1327	1256	1454	1489	1714	1331
50	1479	1435	1365	1387	1193	1239

Untuk menampilkan hasil pengujian secara global, kami memetakan kualitas solusi *S PGA* dan *GA* pada Gambar 5.3. Untuk setiap kombinasi jumlah CPU dan jumlah tugas, kami menghitung rata-rata waktu eksekusi tugas solusi akhir untuk *S PGA* dan *GA*. Hasilnya diplot pada Gambar. 5.3. Sumbu *X* pada bagan mewakili kombinasi hitungan CPU *Pc* dan hitungan tugas *R*. Ini dalam format *(Pc, R).* Sumbu *Y* mewakili waktu eksekusi tugas dalam detik. Gambar 5.3 menunjukkan Tga lebih besar dari Tspga pada semua kasus uji. *Solusi S PGA* menggunakan waktu eksekusi lebih sedikit daripada solusi *GA*.

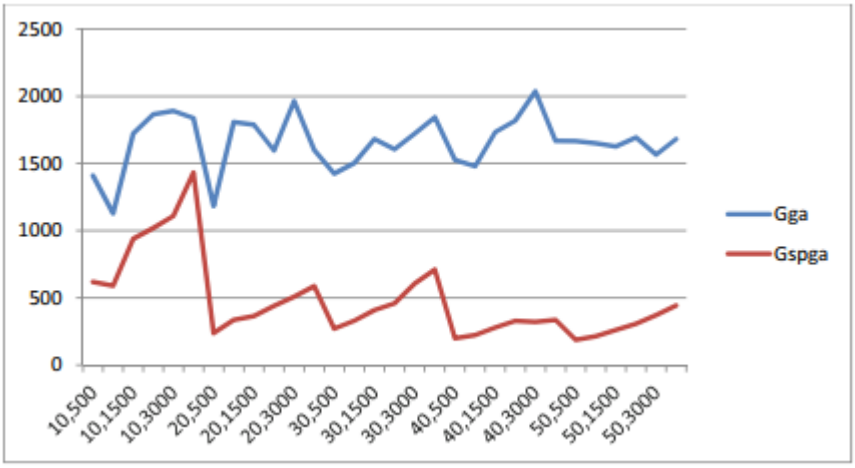


Gambar 5.3. Perbandingan S PGA, Kualitas Solusi GA

Gambar 5.4 dan 5.5 membandingkan kecepatan konvergensi *S PGA* dan *GA.* Sumbu X sama dengan Gambar 5.3 yang merepresentasikan kombinasi hitungan CPU *Pc* dan hitungan tugas *R* dalam bentuk *(Pc, R).* Pada Gambar 5.4, sumbu Y menunjukkan jumlah generasi evolusi yang digunakan oleh algoritma. Ini menunjukkan *GA* menggunakan lebih banyak generasi daripada *S PGA* untuk mencapai solusi yang setara. Sumbu Y menunjukkan waktu pencarian algoritma (detik) pada Gambar 5.5. Dalam semua kasus, *GA* menggunakan lebih banyak waktu daripada *S PGA* untuk mencapai solusi yang setara.

Gambar 5.4 menunjukkan *S PGA* menggunakan lebih sedikit generasi evolusi daripada *GA*. Karena *S PGA* menggunakan komputasi tambahan untuk harga bayangan, setiap operasi mungkin memakan waktu sedikit lebih lama. Dengan demikian, generasi yang lebih sedikit mungkin tidak berkorelasi langsung dengan kecepatan pencarian yang lebih cepat. Gambar 5.5 menunjukkan bahwa *S PGA* menggunakan waktu pencarian yang lebih sedikit daripada *GA.* Ini membuktikan bahwa menghabiskan sedikit waktu ekstra untuk menghitung harga bayangan dapat menghasilkan peningkatan kecepatan yang signifikan. Gambar 5.4 dan 5.5 bersama-sama menunjukkan bahwa *S PGA* menggunakan lebih sedikit waktu dan lebih sedikit generasi evolusi daripada *GA* untuk mencapai solusi optimal.

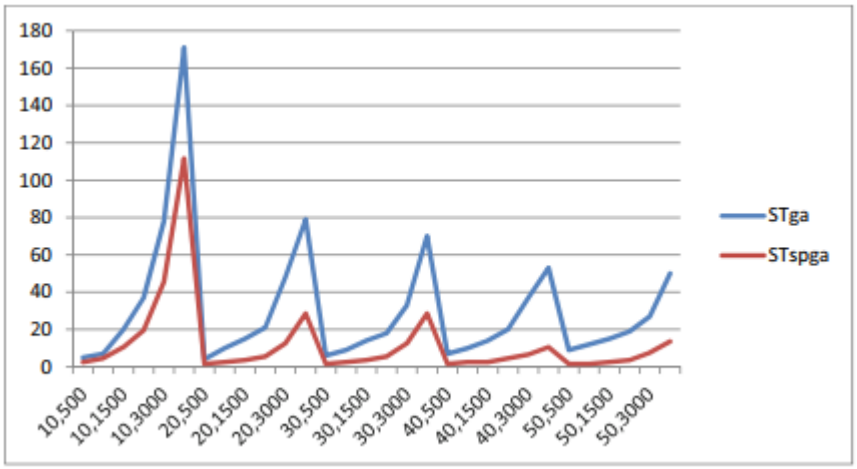
Semua data pengujian dan penelitian menunjukkan bahwa jadwal akhir dari *S PGA* menggunakan lebih sedikit waktu untuk menyelesaikan semua tugas daripada jadwal akhir dari *GA. S PGA* mencapai solusi yang lebih baik daripada *GA. S PGA* juga menggunakan lebih sedikit waktu dan lebih sedikit generasi evolusi daripada *GA* untuk mencapai solusi optimal. Secara keseluruhan, *S PGA* menemukan hasil yang lebih baik dari *GA* dan lebih cepat dari *GA.*



Gambar 5.4. Perbandingan *S PGA,* Kecepatan Pencarian *GA* menggunakan Hitungan Generasi

5.5 KESIMPULAN

GA adalah algoritma pencarian global berbasis populasi efektif yang telah memecahkan banyak masalah optimisasi. Banyak algoritma pencarian memiliki persyaratan yang ketat. Pemrograman linier hanya dapat mengoptimalkan masalah yang dapat direpresentasikan dalam format linier. *GA* menimbulkan sedikit batasan untuk masalah ini. Itu dapat mengoptimalkan banyak masalah yang memiliki fungsi dan operasi tujuan yang jelas.



Gambar 5.5. Perbandingan *S PGA*, Kecepatan Pencarian *GA* menggunakan Waktu

Tantangan dari *GA* serbaguna adalah kinerjanya. Untuk masalah kompleks yang besar, *GA* berbasis swam membutuhkan perhitungan yang besar, membutuhkan waktu lama untuk konvergen, dan dapat memberikan solusi yang kurang optimal.

Kontrol evolusi adalah kunci kinerja *GA*. Dalam proses pencarian, operasi *GA* menghasilkan banyak solusi secara acak dan pemfilteran elit menggerakkan pencarian menuju solusi optimal. Nilai fitness digunakan untuk memfilter kandidat solusi. Menghasilkan dan memfilter banyak solusi acak menghabiskan banyak sumber daya komputasi dan waktu pencarian. Oleh karena itu, kami memperkenalkan harga bayangan untuk menghasilkan solusi yang lebih baik.

Kami menggunakan harga bayangan untuk merepresentasikan potensi kontribusi komponen terhadap solusi. Ini digunakan untuk membandingkan komponen dan operasi *GA* langsung. Harga bayangan dapat direpresentasikan dalam berbagai bentuk, harga material sebenarnya atau keadaan relatif. Dalam artikel ini, ini adalah prosedur yang tertanam dalam operasi.

S PGA baru mengimplementasikan sistem dua pengukuran untuk meningkatkan algoritma. Nilai kebugaran mengevaluasi solusi keseluruhan dan mengembangkan pencarian menuju solusi optimal. Harga bayangan mengevaluasi komponen dan membantu menghasilkan solusi yang lebih baik. *S PGA* untuk penjadwalan ramah lingkungan memberikan hasil yang sangat baik. Dalam semua kasus uji dan studi, *S PGA* menghasilkan solusi yang lebih baik dan konvergen lebih cepat.

BAB 6
MANAJEMEN TERMAL DI BANYAK SISTEM INTI

Kepadatan daya yang tinggi dan suhu pengoperasian dalam sistem multi-prosesor menimbulkan sejumlah efek yang tidak diinginkan, termasuk penurunan kinerja, biaya operasional dan pendinginan yang tinggi, dan penurunan keandalan yang menyebabkan kegagalan sistem. Sistem banyak-inti membawa peluang menarik dalam desain dan manajemen sistem karena banyaknya paralelisme perangkat keras, sambil memperkenalkan tantangan baru karena kompleksitasnya dan beban kerja yang sangat beragam yang diharapkan dapat dijalankan pada sistem ini. Pemantauan dan manajemen termal yang efisien, merancang arsitektur yang menyadari termal, dan optimalisasi parameter multi-level dapat mengurangi beberapa efek termal yang tidak diinginkan sekaligus mempertahankan kinerja dan tingkat energi yang diinginkan. Secara khusus, teknik ini membantu dalam evolusi sistem komputasi hijau. Bab ini memberikan diskusi kualitatif tentang teknik manajemen termal untuk sistem banyak-inti. Kami akan menjelaskan pertanyaan-pertanyaan berikut secara rinci: Apa saja tantangan desain khusus dalam memantau suhu sistem skala besar? Bagaimana kita dapat mengeksploitasi pengoptimalan multi-level saat runtime sebagai respons terhadap perilaku dinamis dari beban kerja prosesor? Bagaimana beban kerja yang muncul memengaruhi distribusi termal pada sistem banyak-inti?

6.1 PENDAHULUAN

Sistem banyak inti menawarkan potensi untuk menghasilkan 10 hingga 100 kali kapasitas pemrosesan dibandingkan dengan sistem inti tunggal tradisional. Diamati dalam prototipe awal bahwa dengan mengintegrasikan sejumlah inti yang lebih sederhana pada satu die, teknologi chip banyak inti ini memiliki beberapa keunggulan termasuk tingkat integrasi. Penskalaan catu daya yang tidak ideal dan densitas daya yang meningkat (W/cm2) adalah salah satu tantangan yang menyertai penskalaan.

Kepadatan daya yang tinggi menimbulkan pembangkitan panas dalam jumlah besar. Misalnya, bahkan ketika satu inti menghilangkan daya 1W, daya keseluruhan yang hilang akan menjadi sekitar 100W untuk sistem 100 inti. Fakta ini, dikombinasikan dengan heterogenitas areal konsumsi daya di sirkuit terpadu banyak-inti serta difusi panas lateral yang relatif lambat dalam silikon menghasilkan hot spot termal lokal. Akibatnya, korelasi yang kuat antara suhu dan berbagai mekanisme kegagalan chip sangat menurunkan keandalan dan umur chip. Faktanya, perbedaan kecil pada suhu pengoperasian (yaitu, 10 ◦C - 15 ◦C) dapat menghasilkan perbedaan 2X dalam masa pakai perangkat ini. Borkar memperkirakan bahwa kebutuhan kemasan termal meningkatkan total biaya per keping sebesar $1/W, ketika daya keping adalah 35-40W. Oleh karena itu, sangat penting untuk memantau, mengelola/mengontrol suhu pada chip untuk memaksimalkan masa pakai perangkat dan memastikan kebenaran komputasi.

Dalam platform banyak-inti, hot spot chip sangat bergantung pada beban kerja. Untuk memaksimalkan kinerja dan keandalan secara bersamaan pada perangkat ini, tugas dan sumber daya harus dikelola dengan cara yang sadar termal sehingga suhu tidak melebihi nilai

ambang batas kritis dan gradien termal diminimalkan. Jadi, secara umum, tujuan manajemen termal adalah untuk memaksimalkan kinerja sistem (atau mempertahankan kinerja yang diinginkan) sambil meminimalkan hot spot dan gradien suhu. Manajemen termal dalam sistem banyak-inti dapat diatur ke dalam tiga fase berikut: (1) Pemantauan termal, yang menangani penempatan sensor termal dan pemantauan dinamis profil termal sistem untuk memberikan umpan balik ke unit manajemen termal runtime; (2) prediksi suhu, yang menangani pemodelan dan perkiraan perilaku termal runtime untuk desain dan manajemen sistem yang efisien; dan (3) manajemen termal runtime, yang merupakan seperangkat teknik untuk mengurangi efek suhu yang berbahaya di bawah batasan energi dan/atau kinerja. Bab ini membahas masing-masing dari ketiga langkah ini secara rinci.

6.2 PEMANTAUAN TERMAL

Penskalaan teknologi telah memungkinkan integrasi banyak inti pada satu cetakan. Kecenderungan densitas daya yang tinggi ini telah menyebabkan suhu on-chip yang meningkat secara keseluruhan dan area lokal dengan suhu yang jauh lebih tinggi, disebut sebagai hot spot. Gradien termal lintas chip telah mencapai setinggi 50 ◦C, dan nilai ini meningkat dengan frekuensi operasi yang lebih tinggi. Temperatur tinggi dan variasi termal yang besar di seluruh chip menimbulkan serangkaian bahaya keandalan yang dapat menyebabkan fungsionalitas sirkuit yang tidak diharapkan dan melemahkan parameter fisik chip. Untuk mengurangi efek termal seperti itu, berbagai manajemen termal dinamis dan teknik pengoptimalan telah diusulkan. Agar manajemen termal dinamis menjadi efisien, sangatlah penting untuk memiliki penginderaan suhu yang akurat pada chip (misalnya, untuk setiap inti dalam sistem banyak inti) dan meneruskan informasi ini ke unit manajemen termal. Beberapa kelompok penelitian telah mengembangkan algoritma pengoptimalan yang menghasilkan penggunaan jumlah sensor minimum sambil mempertahankan cakupan yang memadai. Algoritme pengoptimalan ini terbagi dalam dua kategori utama: penempatan sensor yang seragam dan tidak seragam.

Penempatan Sensor Seragam

Skema pengoptimalan penempatan sensor yang seragam dimaksudkan untuk digunakan dengan chip yang memiliki pola termal khas yang tidak diketahui. Sensor ditempatkan dalam jaringan statis yang seragam di seluruh chip dengan maksud untuk dapat mendeteksi semua pelanggaran suhu, di mana pun terjadi pada chip. Seperti disebutkan sebelumnya, hanya jaringan sensor berbutir halus yang mampu mencapai akurasi hampir sempurna. Karena batasan biaya yang signifikan terkait dengan overhead sensor, granularity grid harus dibatasi, sehingga membatasi akurasi model ini.

Pendekatan interpolasi linier lurus ke depan untuk mmperhitungkan pembatasan ini dan menyempurnakan pengukuran suhu. Skema interpolasi dengan 4×4 grid sensor untuk meningkatkan grid seragam statis dengan ukuran yang sama tanpa interpolasi dengan rata-rata 1,59◦C melintasi benchmark SPEC2000 dalam prosesor *single-core.*

Salah satu keuntungan menerapkan teknik alokasi sensor termal yang seragam adalah tidak bergantung pada data profil termal. Tidak ada pengetahuan tentang lokasi hot spot dan suhu yang perlu diperoleh sebelum menerapkan teknik jenis ini. Karakteristik ini,

bagaimanapun, membatasi keakuratan model grid seragam karena jarak antara lokasi sensor dan hot spot tidak dapat diminimalkan. Tanpa pengetahuan tentang peta termal umum yang dihasilkan, sensor yang diatur dalam kisi yang seragam tidak akan mampu mendeteksi titik panas seakurat jumlah sensor yang sama yang terletak di dekat titik panas umum.

Penempatan Sensor Tidak Seragam

Skema pengoptimalan penempatan sensor yang tidak seragam dimaksudkan untuk digunakan di mana peta termal dari eksekusi chip tipikal di beberapa aplikasi tersedia untuk analisis. Jenis teknik ini memanfaatkan hot spot yang diketahui pada chip untuk menentukan lokasi yang paling menguntungkan untuk penempatan sensor. Pendekatan yang naif adalah menempatkan sensor di setiap hot spot yang ditemukan melalui profil termal di beberapa aplikasi. Sayangnya, pendekatan ini tidak praktis karena jumlah hot spot yang tinggi sangat mungkin terjadi, dan penggunaan sensor dalam jumlah besar juga tidak praktis. Idealnya, sejumlah minimum sensor akan diatur pada chip sedemikian rupa untuk memberikan cakupan semua kemungkinan hot spot. Hot spot tidak akan selalu berada di lokasi yang sama pada chip selama eksekusi satu program, dan berbagai aplikasi yang berjalan pada chip yang sama akan menampilkan hot spot di wilayah yang berbeda. Lokasi hot spot dan suhu bergantung pada aplikasi, dan tidak mungkin solusi yang dioptimalkan untuk satu aplikasi akan cukup untuk beban kerja lainnya. Satu konfigurasi penempatan sensor harus cukup untuk semua hot spot yang mungkin muncul selama eksekusi program apa pun.

Beberapa metode yang mendeteksi pelanggaran termal dengan jumlah sensor yang terbatas telah dikembangkan berdasarkan lokasi hot spot dan suhu yang ditemukan melalui profiling termal. Radius maksimum R antara titik panas dan lokasi sensor termal potensial, sembari membatasi kesalahan hingga derajat ΔT. Nilai ΔT menunjukkan perbedaan antara nilai suhu maksimum dan minimum dalam chip. Dalam persamaan ini, K digunakan untuk mewakili efek dari bahan yang termasuk dalam chip. Parameter ini meliputi ketebalan die paket prosesor, penyebar panas, dan bahan antarmuka termal dikalikan dengan faktor resistivitas termal khusus untuk setiap bahan.

$$R = 0.5 \cdot K \cdot \ln\left(\frac{T_{max}}{T_{max} - \Delta T}\right) \qquad (6.1)$$

Pengelompokan Ambang-Kualitas

Persamaan 6.1 dengan algoritma pengelompokan ambang kualitas (QT) yang biasa digunakan dalam pengelompokan gen. Memperlakukan hot spot sebagai titik yang harus dikelompokkan, pengelompokan hot spot dan lokasi sensor yang sesuai ditentukan berdasarkan nilai *Tmax* untuk semua hot spot di masing-masing cluster. Pengelompokan *QT* adalah teknik berulang yang menetapkan hot spot ke cluster berdasarkan lokasi fisiknya pada chip relatif terhadap hot spot lainnya. Lokasi sensor untuk setiap cluster disempurnakan setelah penambahan kandidat hot spot untuk menjadi pusat massa dari hot spot yang disertakan, sehingga mendapatkan kemungkinan lokasi sensor terbaik untuk kumpulan titik data hot spot yang diberikan. Titik panas yang baru ditambahkan akan disimpan di klaster ini hanya jika setiap titik panas lainnya di klaster terletak dalam jarak *R* dari pusat klaster.

Penempatan 23 sensor dengan Tmax = 3◦C menggunakan teknik QT clustering pada inti prosesor Alpha 21364 dan hot spot yang dihasilkan oleh benchmark SPEC2000. 23 sensor merasakan profil termal lengkap inti dengan kesalahan rata-rata 0,2899◦C.

Meskipun algoritma pengelompokan terbukti cukup untuk memantau kejadian termal, ia tidak memasukkan jumlah kluster atau sensor yang tersedia untuk digunakan untuk desain tertentu. Ini bisa merugikan karena beberapa alasan. Algoritma tidak mengakhiri eksekusi hingga setiap titik panas ditempatkan dalam sebuah kluster, membuat kluster baru jika diperlukan untuk menyertakan titik panas yang terletak jauh dari yang lain. Jumlah sensor yang dibutuhkan oleh algoritma pengelompokan QT mungkin tidak tersedia untuk digunakan dalam desain praktis. Untuk menempatkan lebih sedikit sensor menggunakan pengelompokan QT, hot spot yang dialokasikan ke nilai jarak sensor harus ditingkatkan, yang dapat menurunkan akurasi hasil seluruh model.

Pengelompokan K-Means

Algoritma *k-means* clustering dasar membutuhkan sejumlah sensor untuk ditempatkan sebagai parameter input, *k.* Titik panas ditempatkan ke dalam k cluster yang berbeda, dengan sensor suhu ditempatkan di pusat setiap cluster. Penetapan cluster dipilih sedemikian rupa sehingga jarak kuadrat rata-rata dari setiap hot spot ke pusat cluster terdekat diminimalkan. Pertama, pusat k cluster dipilih secara acak dari kumpulan titik hot spot yang diketahui. Setiap hot spot kemudian ditugaskan ke sebuah cluster *Cj* sehingga jarak Euclidean *E(Oj, hi)* antara hot spot hi dan pusat cluster ini *Oj* diminimalkan. Persamaan untuk menentukan jarak Euclidean antara dua titik ditunjukkan pada Persamaan 6.2. Dalam persamaan ini*, (hix,hiy*) merepresentasikan lokasi hot spot *hi* dan *(Ojx,Ojy)* merepresentasikan lokasi pusat cluster *Oj* pada bidang *(x,y).*

$$E(O_j, h_i) = (O_{jx} - h_{ix})^2 + (O_{jy} - h_{iy})^2 \qquad (6.2)$$

Di akhir setiap iterasi, setiap pusat klaster diperbarui menjadi pusat massa dari lokasi semua hot spot yang ditetapkan ke klaster tersebut. Jarak Euclidean antara hot spot dan pusat cluster kemudian dihitung ulang. Jika jarak minimum baru antara titik panas dan pusat klaster yang berbeda ditemukan, titik panas dipindahkan ke klaster yang sesuai. Proses ini diulang sampai tidak ada hot spot yang dipindahkan ke cluster yang berbeda atau jumlah total semua jarak Euclidean tidak mengalami peningkatan yang signifikan.

Versi thermal gradient-aware dari algoritma k-means clustering. Tujuan utama dari pendekatan ini adalah menempatkan sensor suhu ke hot spot yang biasanya memiliki suhu lebih tinggi. Cluster dibentuk dalam ruang 3-D dengan menggunakan suhu setiap hot spot, t, sebagai dimensi ketiga untuk menghitung jarak Euclidean, seperti yang ditunjukkan pada Persamaan 6.3.

$$E(O_j, h_i) = (O_{jx} - h_{ix})^2 + (O_{jy} - h_{iy})^2 + (O_{jt} - h_{it})^2 \qquad (6.3)$$

Pusat cluster diperbarui di setiap iterasi dengan pertimbangan suhu hot spot. Hot spot dibobotkan dalam perhitungan centroid relatif terhadap besarnya suhu mereka. Seperti dalam

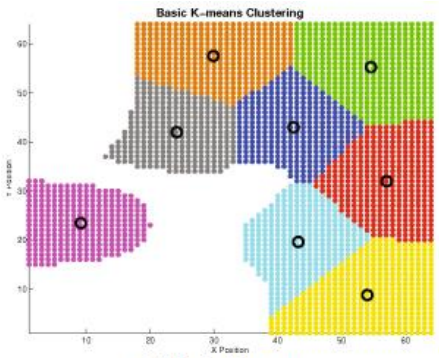
algoritme pengelompokan k-means dasar, pusat klaster dan penugasan klaster hot spot disempurnakan secara iteratif hingga tidak ada hot spot yang dipindahkan ke klaster yang berbeda atau jumlah total semua jarak Euclidean 3-D tidak memiliki peningkatan yang signifikan.

Clustering k-means kesadaran gradien termal menghasilkan kesalahan rata-rata terbaik 2,10◦C menempatkan 16 sensor pada satu inti dan kesalahan rata-rata 1,63◦C menggunakan 36 sensor. Menggunakan dua nomor sensor yang sama dengan data profil termal yang identik, algoritme pengelompokan k-means dasar menghasilkan kesalahan rata-rata 4,58◦C dan 3,05◦C. Ini menunjukkan peningkatan masing-masing 2,48◦C dan 1,42◦C. Semua eksperimen dijalankan menggunakan HotSpot yang dikonfigurasi untuk Alpha 21364, dan posisi hot spot ditentukan dari pola termal yang berkaitan dengan tolok ukur SPEC2000.

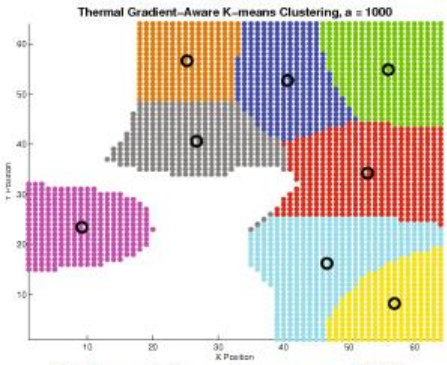
Meskipun k-means clustering sadar gradien termal bekerja dengan baik dalam banyak kondisi, teknik ini tidak selalu optimal dalam skenario distribusi hot spot yang kompleks dan dapat menghasilkan solusi yang lebih buruk daripada pendekatan k-means dasar. Dalam kasus seperti itu, hot spot sering diurutkan ke dalam kelompok yang tidak sesuai karena kesamaan suhu dan terlepas dari lokasi fisiknya pada chip. Gambar 6.1(a) menunjukkan hasil clustering k-means dengan delapan sensor. Meskipun pengelompokan hot spot sedikit berbeda untuk kedua metode tersebut, lokasi penempatan sensornya hampir identik. Untuk membuat perbedaan dalam penempatan sensor, suhu hot spot yang digunakan dalam perhitungan k-means kesadaran gradien termal dapat diskalakan menurut Persamaan 6.4, di mana a adalah konstanta yang menentukan kecuraman gradien suhu.

$$New\,Temperatures = a \cdot \frac{Original\,Temperatures}{Maximum\,Temperature} \qquad (6.4)$$

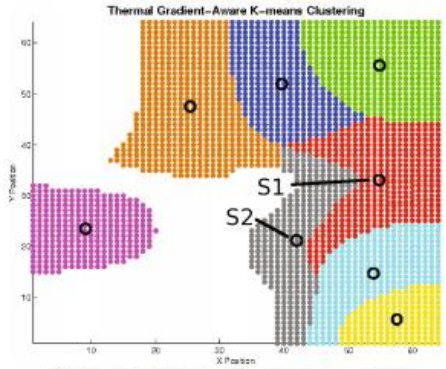
Gambar 6.1(b) dan 6.1(c) menunjukkan hasil pengelompokan menggunakan Persamaan 6.4 dengan a = 1000 dan a = 2500, masing-masing, pada set hot spot yang sama yang digunakan pada Gambar 6.1(a). Dalam kedua situasi tersebut, sensor ditempatkan lebih dekat ke titik panas bersuhu lebih tinggi dan lebih jauh dari titik panas bersuhu lebih rendah. Namun, banyak penugasan klaster hot spot tidak sesuai secara spasial. Pada Gambar 6.1(c), misalnya, banyak hot spot merah yang dikelompokkan dengan sensor S1 akan lebih tepat dikelompokkan dengan hot spot abu-abu yang dikelompokkan dengan sensor S2, dan sebaliknya.



(a) Basic K-means



(b) Thermal-Aware K-means, a = 1000



(c) Thermal-Aware K-means, a = 2500

Gambar 6.1. K-means variasi penempatan sensor pengelompokan.

Meskipun k-means clustering sadar gradien termal efektif untuk prosesor single-core, itu tidak sesuai untuk prosesor multi-core dengan interaksi termal antar-core yang kuat karena kemungkinan hot spot tidak mungkin terjadi. muncul di lokasi yang sama di setiap inti.

Menentukan Titik Panas Termal untuk Membantu Alokasi Sensor

Ada banyak properti yang perlu dipertimbangkan saat menentukan lokasi mana pada chip yang dianggap sebagai hot spot. Memvariasikan aturan penentuan memiliki potensi untuk secara signifikan mempengaruhi lokasi penempatan sensor yang dihasilkan. Pertukaran antara karakterisasi peta termal penuh dan deteksi hot spot harus dipertimbangkan saat mengidentifikasi lokasi hot spot awal.

Hot Spot Lokal

Salah satu aturan penentuan yang umum adalah menetapkan sejumlah hot spot tertentu per blok fungsional di dalam inti pemrosesan, disebut sebagai hot spot lokal. Teknik ini mendorong penempatan sensor di seluruh cetakan dan paling sesuai untuk mengkarakterisasi peta termal lengkap prosesor. Lokasi hot spot akan menjadi titik yang mencapai suhu maksimum lokal untuk setiap blok fungsional. Untuk banyak blok fungsional, titik terpanas akan berada di dekat tepi blok yang berdekatan dengan komponen yang lebih panas.

Hot Spot Global

Metode penentuan hot spot kedua adalah merekam hot spot global, atau lokasi mana pun pada die yang mencapai atau melampaui ambang suhu darurat yang ditentukan, biasanya mendekati 82◦C atau 355 K. Sensor suhu akan ditempatkan lebih dekat ke lokasi pada chip dengan suhu yang jauh lebih tinggi, dan dengan demikian

kemungkinan besar tidak akan tersebar di seluruh chip. Teknik ini paling baik untuk mengenali suhu darurat dengan cepat daripada memulihkan peta termal penuh seluruh prosesor. Selain itu, jumlah hot spot yang ditentukan dengan teknik ini berkorelasi terbalik dengan ambang batas suhu darurat yang ditentukan. Ambang batas yang lebih tinggi akan menghasilkan lebih sedikit hot spot yang semuanya dapat ditempatkan dalam satu blok fungsional di dalam prosesor. Alternatifnya, ambang batas yang cukup rendah akan menghasilkan banyak hot spot, yang berpotensi menutupi lebih dari 50% prosesor. Sejumlah besar hot spot akan memungkinkan sensor tersebar melalui area yang lebih luas. File register integer berulang kali merupakan komponen terpanas di inti Alpha 21364 di seluruh benchmark SPEC2000. Memilih ambang batas suhu darurat yang tinggi dapat mengakibatkan penempatan sensor hanya dalam file register bilangan bulat, sedangkan ambang batas yang lebih rendah akan memungkinkan sensor ditempatkan di blok fungsional yang berdekatan di seluruh prosesor. Pertukaran antara deteksi hot spot cepat dan pemulihan peta termal penuh harus dipertimbangkan.

Subsampling Peta Termal yang Tidak Seragam

Dalam arsitektur banyak-inti, ada kemungkinan besar untuk mengukur hot spot global dalam jumlah yang sangat besar. Jumlah titik panas mungkin sangat besar sehingga algoritma pengelompokan tidak dapat menempatkan sejumlah sensor yang memadai di dekat titik terpanas. Untuk mengurangi jumlah titik yang akan dikelompokkan sambil mempertahankan representasi data termal yang jelas, algoritma subsampling yang tidak seragam dapat digunakan untuk mendapatkan subset dari lokasi analisis termal utama pada sebuah chip. Lebih banyak sampel dipilih dari daerah dengan suhu lebih tinggi dan lebih sedikit titik dipilih dari daerah dengan suhu lebih rendah. Subsampling peta termal kemungkinan akan mencapai keseimbangan antara pengukuran suhu yang seragam dan deteksi darurat termal. Setelah peta termal disubsampling, algoritma pengelompokan dapat digunakan pada subsampel untuk menentukan lokasi penempatan sensor.

Dua algoritma subsampling non-seragam berbasis gradien untuk gambar memilih piksel sampel dari gambar yang diberikan sedemikian rupa sehingga daerah gradien konstan akan diwakili oleh sejumlah sampel yang berbanding lurus dengan besarnya gradien.

Subsampling Deterministik

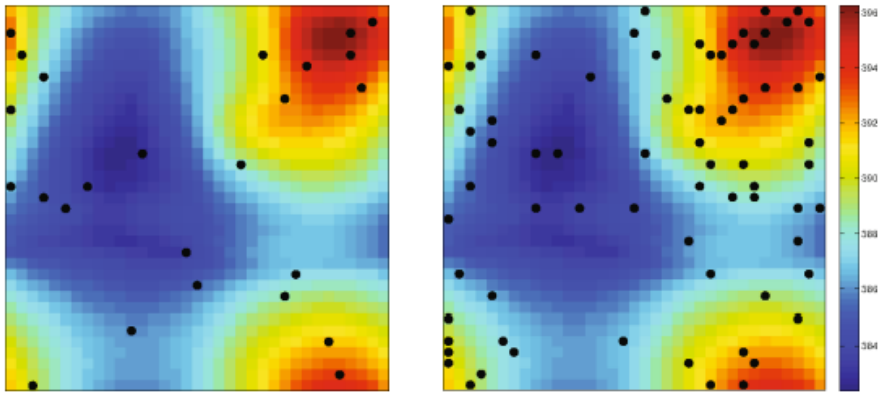
Versi deterministik dari algoritma ini menyatakan bahwa untuk mengkuantisasi kumpulan data *I1* menjadi level *Q*, daftar lokasi piksel *Iq* dengan norma gradien terkuantisasi *q* harus dibuat untuk setiap level *q*. Setelah semua piksel telah didistribusikan ke tingkat kuantisasi yang sesuai, setiap piksel dalam setiap daftar Iq akan dipilih, di mana *sq = c/q* untuk konstanta *c.* Spesifikasi ini memastikan bahwa sampel lebih sering dipilih di wilayah dengan gradien tinggi. Nilai konstanta *c* disesuaikan untuk menghasilkan jumlah sampel yang lebih besar atau lebih kecil.

Menerapkan algoritma subsampling ke peta termal lengkap dari inti prosesor sembari memperlakukan nilai suhu sebagai gradien menghasilkan lebih banyak sampel yang lebih dekat ke wilayah dengan lebih banyak hot spot dan lebih sedikit sampel di dekat wilayah yang lebih dingin. Menyesuaikan nilai c mempengaruhi jumlah sampel

yang diambil dari peta termal. Sebelum melakukan subsampling, suhu harus dinormalisasi menurut Persamaan 6.5 menggunakan suhu minimum Tmin dan suhu maksimum Tmax dalam peta termal yang digunakan untuk analisis.

$$I_{norm} = \frac{\|\nabla I_1\| - T_{min}}{T_{max} - T_{min}} \tag{6.5}$$

Melakukan algoritma subsampling non-seragam deterministik pada peta termal sampel menghasilkan hasil sampel yang ditampilkan pada Gambar 6.2(a) dan 6.2(b). Kedua proses tersebut menggunakan 25 level kuantisasi. Gambar 6.2(a) menunjukkan hasil dari pengaturan konstanta c = 5. Konstanta ini menghasilkan sampel yang jauh lebih sedikit daripada pengaturan konstanta c = 0,25 seperti yang ditunjukkan pada Gambar 6.2(b). Kedua plot mengungkapkan lokasi pengambilan sampel yang tersebar di seluruh peta termal. Algoritma ini cukup konservatif dan mirip dengan sampling seragam.



(a) Thermal deterministic subsamples for $c = 5$ (b) Thermal deterministic subsamples for $c = 0.25$

Gambar 6.2. Hasil subsampling non-seragam deterministik termal dengan 25 tingkat kuantisasi.

Subsampling Stokastik

Versi stokastik dari pengambilan sampel tidak seragam melihat setiap lokasi piksel individual (i, j) dan memutuskan apakah akan memilih piksel ini sebagai sampel atau tidak. Piksel dipilih dengan probabilitas $p(i,j) = min(\alpha * ||\nabla l1||\ (i,j), 1)$. Menyesuaikan konstanta proporsionalitas α menghasilkan sampel yang lebih sedikit atau tambahan.

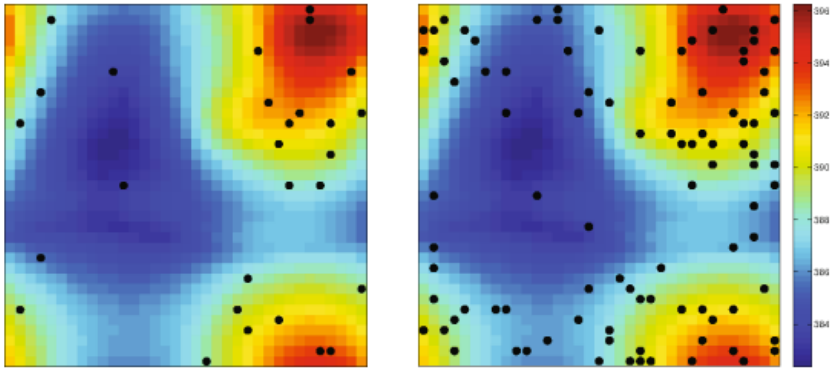
Melakukan algoritma subsampling non-seragam stokastik pada peta termal sampel menghasilkan hasil sampel yang ditampilkan pada Gambar 6.3. Gambar 6.3(a) menunjukkan hasil dari pengaturan konstanta α = 1.5 yang menghasilkan sampel lebih sedikit dibandingkan dengan pengaturan konstanta α = 5 seperti yang ditunjukkan pada Gambar 6.3(b). Kedua plot menunjukkan banyak titik pengambilan sampel di wilayah terpanas, dan hanya satu atau dua sampel di wilayah terdingin. Sampel yang

diambil dalam algoritma subsampling stokastik ini secara akurat mencerminkan gradien termal dari chip.

Pengelompokan K-means dapat digunakan untuk menentukan lokasi sensor yang tepat saat memperlakukan sampel sebagai titik yang akan dikelompokkan. Karena lokasi sampel, penempatan sensor termal yang dihasilkan akan mampu menjaga keseimbangan antara profil seluruh inti dan mendeteksi keadaan darurat termal.

6.3 TEKNIK PEMODELAN DAN PREDIKSI SUHU

Temperatur merupakan faktor utama dalam menentukan performa, keandalan, dan konsumsi daya bocor prosesor modern. Akibatnya, memperkirakan suhu secara akurat merupakan langkah penting dalam manajemen termal yang sukses. Bagian ini pertama membahas prinsip dan alat pemodelan termal. Tujuan dari pemodelan termal adalah untuk memberikan penilaian suhu yang akurat untuk membantu desain sistem dan memungkinkan evaluasi skenario runtime.



(a) Thermal stochastic subsamples for $\alpha = 1.5$ (b) Thermal stochastic subsamples for $\alpha = 5$

Gambar 6.3. Hasil subsampling non-seragam stokastik termal.

Bagian kedua dari bagian ini membahas prediksi termal, yang bergantung pada teknik berbiaya rendah untuk memperkirakan suhu saat ini atau mendatang berdasarkan pembacaan sensor termal atau profil daya/kinerja. Prediksi termal dapat digunakan sebagai peringatan dini dalam kebijakan manajemen termal dinamis prediktif (lihat Bagian 6.4) atau untuk memberikan perkiraan termal yang lebih komprehensif untuk sistem tanpa sensor angka yang memadai. Dalam kedua kasus tersebut, tujuan utama prediksi termal adalah untuk meningkatkan pengambilan keputusan selama manajemen termal.

Pemodelan Termal

Ada beberapa metode pengukuran suhu pada level chip. Pengumpulan data suhu dengan menggunakan metode kontak titik (termokopel) dibatasi oleh banyaknya titik yang akan dipantau dan kecilnya ukuran komponen. Menghubungkan puluhan atau ratusan termokopel sangat memakan waktu. Pencitraan termal inframerah (IR) adalah teknik baru yang mengatasi masalah ini dengan menyediakan peta dua dimensi (2-D) yang komprehensif dari ribuan suhu dalam hitungan detik. Ini dilakukan tanpa perlu melakukan kontak dengan

komponen. Pendekatan ini, bagaimanapun, mahal dan memakan waktu dan hanya dapat diterapkan setelah desain. Akhirnya, sensor termal on-chip juga menyediakan pembacaan suhu; namun, jumlah sensor yang dapat digunakan pada sebuah chip dibatasi oleh batasan desain chip seperti area dan biaya (teknik penempatan sensor yang efisien diulas di Bagian 6.2). Oleh karena itu, penting untuk memiliki model termal chip penuh dan alat simulasi yang dapat memberikan profil suhu cetakan.

Pada sebuah chip, panas dihasilkan baik di substrat maupun di interkoneksi. Disipasi daya tambahan dihasilkan dari pemanasan Joule (atau pemanasan sendiri) yang disebabkan oleh aliran arus dalam jaringan interkoneksi. Meskipun pemanasan Joule interkoneksi hanya merupakan sebagian kecil dari total disipasi daya dalam chip, kenaikan suhu pada interkoneksi akibat pemanasan Joule dapat menjadi signifikan. Ini karena fakta bahwa interkoneksi biasanya terletak jauh dari heat sink oleh beberapa lapisan bahan isolasi yang memiliki konduktivitas termal lebih rendah daripada silikon.

Sumber utama pembangkitan panas adalah disipasi daya perangkat yang tertanam di substrat. Temperatur pengoperasian chip VLSI dapat dihitung dari persamaan linier berikut:

$$T_{chip} = T_a + R_\theta \frac{P_{tot}}{A} \qquad (6.6)$$

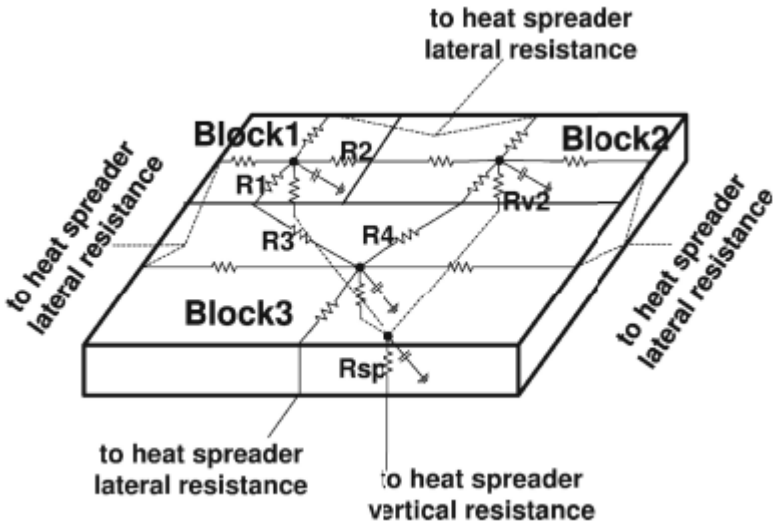
di mana *Tchip* adalah suhu chip rata-rata (persimpangan Si), *Ta* adalah suhu sekitar, *Ptot* (dalam W) adalah konsumsi daya total, *A* (dalam cm2) adalah area chip, dan *Rϑ* adalah resistansi termal setara dari lapisan substrat (Si) ditambah paket dan heat sink (cm2 degC/W). Jadi, untuk menghitung suhu, kita perlu menghitung atau mengukur konsumsi daya, membuat model termal chip untuk menghitung resistansi termal, dan memiliki informasi tentang lingkungan (yaitu, suhu sekitar).

Konsumsi daya chip VLSI terdiri dari daya dinamis, daya hubung singkat, dan daya statis (atau kebocoran), di mana daya dinamis dan daya bocor mendominasi prosesor saat ini. Pemodelan dan pengukuran daya adalah area yang terhubung erat dengan pemodelan termal. Praktik pengukuran daya arus melibatkan penggunaan sensor arus dan pengukur daya untuk pengukuran tingkat chip atau server, atau menggunakan sensor tegangan dan arus pada chip, jika tersedia. Alat pemodelan daya prosesor mencakup model tingkat arsitektur seperti yang dapat dihubungkan ke simulator kinerja yang akurat instruksi, atau model tingkat sirkuit yang tersedia dalam alat desain/analisis sirkuit yang umum digunakan.

Sementara Persamaan 6.6 menghitung suhu seluruh chip, menghitung suhu blok individu (misalnya, unit atau inti mikroarsitektur) atau sel grid berbutir halus membutuhkan pembuatan model termal untuk mensimulasikan perpindahan panas di antara unit-unit pada chip juga. sebagai aliran panas dari persimpangan ke udara.

Penelitian dari HotSpot alat pemodelan termal otomatis untuk membangun jaringan R-C termal yang diberikan properti paket, properti chip, dan tata letak chip. Alat ini mampu memberikan estimasi respons suhu tunak dan transien untuk jejak konsumsi daya yang diberikan. Gambar 6.4 menunjukkan jaringan RC untuk lapisan chip. Penyebar panas dan heat sink juga dimodelkan melalui jaringan serupa.

Untuk mempercepat performa dan simulasi termal, salah satu metodenya adalah meniru eksekusi inti pada platform FPGA dan menjalankan model berbasis jaringan R-C termal otomatis untuk menghitung suhu bersamaan dengan emulasi performa. Perpanjangan terbaru dari model termal otomatis mencakup kemampuan pemodelan untuk sistem bertumpuk 3D dan untuk pendinginan cair berbasis saluran mikro.



Gambar 6.4. Jaringan R-C termal untuk chip

Selain alat simulasi termal ini yang secara khusus menargetkan perkiraan suhu chip, pemecah elemen hingga tujuan umum juga dapat digunakan untuk pemodelan suhu.

Simulator termal berbasis jaringan RC otomatis memberikan akurasi tinggi dalam pemodelan. Model ini, bagaimanapun, memiliki overhead runtime yang cukup besar, mencegahnya untuk dimanfaatkan untuk perkiraan suhu runtime. Untuk alasan ini, beberapa teknik prediksi suhu (atau peramalan) berbiaya rendah telah diusulkan. Selanjutnya, kita membahas metode prediksi termal.

Prediksi Suhu

Menghitung suhu chip menggunakan persamaan linier (Persamaan 6.6) adalah salah satu metode untuk estimasi suhu yang cepat, dan metode ini telah digunakan dalam optimasi desain yang sadar suhu. Metode prediksi suhu lainnya berfokus pada mengekstrapolasi nilai suhu pada chip secara detail berdasarkan pembacaan suhu dari beberapa sensor. Pendekatan ini sangat berguna untuk sistem banyak-inti besar, mengingat jumlah sensor termal yang memadai mungkin tidak tersedia (lihat Bagian 6.2 untuk pembahasan rinci tentang metode penempatan sensor termal).

Temuan dari Sharifi memperkenalkan teknik untuk secara akurat memperkirakan suhu di sewenang-wenang lokasi pada mati berdasarkan pembacaan suhu bising dari sejumlah sensor yang terletak lebih jauh dari lokasi kepentingan. Teknik mereka pertama membangun jaringan RC dari chip offline. Pengurangan urutan model digunakan untuk menghasilkan sistem yang lebih kecil namun akurat untuk model termal. Setelah model termal dibangun, teknik menerapkan pemfilteran Kalman ke model orde tereduksi dari sistem. Kalibrasi

berakhir saat filter mencapai kondisi stabilnya. Filter kondisi-mapan yang dihasilkan digunakan saat runtime untuk melakukan estimasi suhu. Filter Kalman mengestimasi suhu dengan cara prediksi yang benar berdasarkan beberapa nilai sensor suhu dan perkiraan konsumsi daya. Saat runtime, prediktor memproyeksikan suhu menggunakan suhu saat ini. Tahap "pembaruan pengukuran" setelah proyeksi menggabungkan pengukuran baru ke dalam perkiraan apriori untuk mendapatkan perkiraan suhu *a* posteriori yang lebih baik.

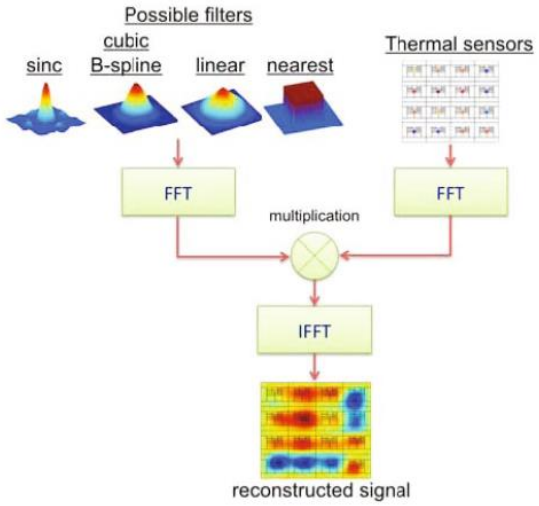
Metode lain untuk estimasi suhu berbutir halus menggunakan sensor termal dalam jumlah terbatas adalah menggunakan teknik analisis Fourier. Secara khusus, penulis menggunakan teori pengambilan sampel Nyquist-Shannon untuk menentukan jumlah sensor termal paling sedikit yang dapat sepenuhnya mencirikan status termal runtime prosesor. Mengingat jumlah sensor termal yang terbatas, teknik ini menerapkan teknik rekonstruksi sinyal yang menghasilkan karakterisasi termal beresolusi penuh dari sebuah prosesor. Gambar 6.5 menunjukkan aliran karakterisasi termal runtime. Teknik pertama mengambil Fast Fourier Transform (FFT) dari sampel suhu dan ruang-domain representasi dari fungsi interpolasi (sinc, tetangga terdekat, kubik B-spline, fungsi linier). Ini kemudian mengalikan representasi domain spektral yang dihasilkan untuk mendapatkan FFT dari sinyal termal 2D yang direkonstruksi. Terakhir, inverse FFT (IFFT) diterapkan untuk mendapatkan karakterisasi termal resolusi penuh dalam domain ruang. Penulis juga mengusulkan metode yang menangani penempatan sensor yang seragam dan tidak seragam; perhatikan bahwa penempatan yang tidak seragam mungkin timbul karena kendala desain dan tata letak.

Alih-alih menggunakan beberapa sensor untuk estimasi detail suhu on-chip, beberapa teknik prediksi memanfaatkan sensor yang ada dan riwayat pengukuran termal untuk memprediksi suhu di masa mendatang. Estimasi termal terperinci berguna dalam manajemen suhu karena kebijakan manajemen dapat membuat keputusan yang lebih tepat dengan menggunakan proyeksi termal berbutir halus. Peramalan suhu, di sisi lain, berfokus pada mempelajari perilaku suhu dan memperkirakan waktu dekat. Dengan cara ini, kebijakan baru dapat dirancang yang secara proaktif membuat keputusan sadar suhu, seperti membongkar inti yang diproyeksikan akan menjadi lebih panas dalam interval prediksi berikutnya.

Perubahan suhu jangka pendek (level aplikasi) menggunakan regresi kuadrat terkecil yang sesuai dengan data sensor termal, dan menghitung perubahan suhu level inti (lebih lambat) berdasarkan suhu aplikasi yang stabil. Estimasi suhu keseluruhan kemudian merupakan penjumlahan tertimbang dari estimasi suhu jangka pendek dan jangka panjang, di mana faktor bobot dipilih secara empiris. Eksperimen prediksi pada SPEC 2006 suite menunjukkan akurasi yang tinggi.

Sementara prediksi berbasis aplikasi melalui least square fit memberikan estimasi yang baik ketika prediktor dilatih dengan karakteristik termal aplikasi, ada beberapa tantangan yang terkait dengan metode ini. Pertama, ketika jarak prediksi (yaitu, berapa langkah atau detik ke depan yang diprediksi) diubah, akurasi prediksi dapat bervariasi secara signifikan. Alasannya adalah, segera setelah kita memprediksi lebih dari beberapa langkah waktu ke depan, suku dengan eksponen terbesar dalam fungsi penyesuaian kuadrat-terkecil mendominasi, dan akurasi prediksi menurun sejak saat itu. Tantangan lainnya adalah adaptasi

runtime: untuk beban kerja yang berubah secara dinamis, kuadrat terkecil rekursif perlu terus memperbarui koefisien model saat data baru tiba (jika tidak, akurasi akan turun).

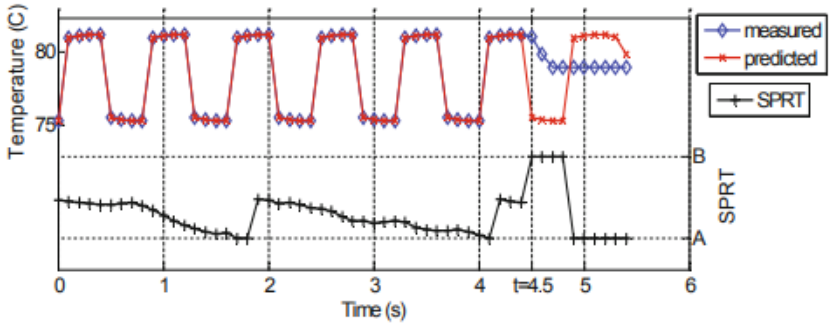


Gambar 6.5. Alur teknik karakterisasi termal menggunakan analisis berbasis FFT

Auto-regressive moving average (ARMA) pemodelan untuk peramalan suhu mengatasi tantangan yang diuraikan di atas melalui konstruksi prediktor menggunakan langkah-langkah berikut. (1) Identifikasi derajat model dan estimasi parameter model. Model ARMA yang pas memiliki kesalahan prediksi akhir yang rendah. (2) Memeriksa model. Kesalahan antara nilai terukur dan prediksi harus didistribusikan secara acak di sekitar nilai rata-rata nol. (3) Adaptasi online menggunakan *Sequential Probability Ratio Test* (SPRT). Tes probabilistik dengan cepat menentukan apakah model yang ada tidak sesuai dengan dinamika suhu beban kerja saat ini dan apakah model baru harus dibangun.

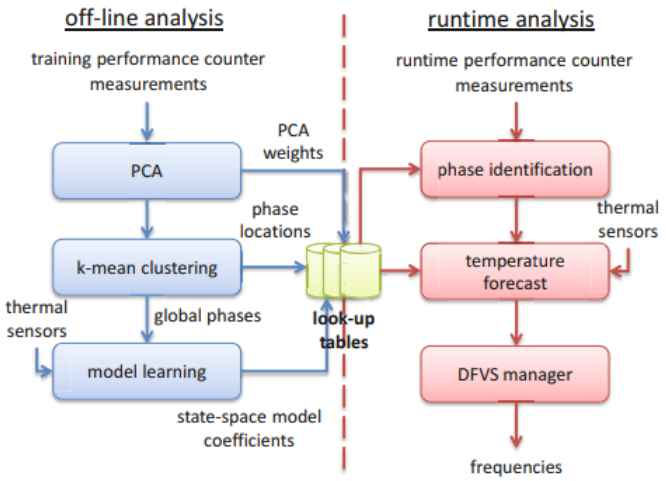
$$y_t + \sum_{i=1}^{p} (a_i\, y_{t-i}) = e_t + \sum_{i=1}^{q} (c_i\, e_{t-i}) \qquad (6.7)$$

Model ARMA*(p,q)* dijelaskan oleh Persamaan 6.7. Dalam persamaan, *yt* adalah nilai deret pada waktu *t* (yaitu suhu pada waktu *t*), *ai* adalah koefisien autoregresif *lag-i, ci* adalah koefisien rata-rata bergerak dan et disebut noise, error atau residual. Residual diasumsikan acak dalam waktu (yaitu, tidak berkorelasi otomatis), dan terdistribusi secara normal. *p* dan *q* masing-masing mewakili urutan autoregressive (AR) dan moving average (MA) dari model.



Gambar 6.6. Prediksi suhu berbasis ARMA dan deteksi online variasi karakteristik termal menggunakan uji rasio kemungkinan

Gambar 6.6 menunjukkan nilai suhu yang diprediksi dan diestimasi menggunakan ARMA. Perhatikan bahwa selama percobaan ini, dinamika suhu berubah, dan SPRT segera mendeteksi perubahan ini (lihat t = 4,5 detik pada gambar). A dan B sesuai dengan ambang pengambilan keputusan SPRT, dihitung berdasarkan persentase alarm salah dan tidak terjawab yang ditentukan pengguna.



Gambar 6.7. Alur pendekatan menggabungkan identifikasi fase beban kerja dan peramalan termal

Pekerjaan terbaru menggunakan pengenalan fase beban kerja selain informasi sensor suhu untuk biaya rendah, prediksi akurasi tinggi. Melalui analisis komponen utama (PCA), dimungkinkan untuk menentukan metrik yang paling relevan untuk mengidentifikasi fase beban kerja yang berbeda. Kemudian, melalui k-means clustering dan model learning, pendekatan tersebut membangun sebuah prediktor secara offline. Saat runtime, overhead dibatasi untuk mengidentifikasi fase melalui pengukuran penghitung kinerja dan perkiraan suhu. Prakiraan ini kemudian dapat digunakan dalam keputusan manajemen termal melalui penyetelan pengaturan frekuensi tegangan. Alur pendekatan ditunjukkan pada Gambar 6.7. Selanjutnya, bab ini membahas teknik pengukuran suhu runtime yang memanfaatkan berbagai tingkat estimasi dan peramalan suhu.

6.4 MANAJEMEN TERMAL WAKTU KERJA

Teknik manajemen termal dinamis (DTM) yang canggih beradaptasi dengan perubahan beban kerja dan lingkungan aplikasi untuk mencapai kinerja yang paling kuat. Secara umum, teknik manajemen termal adaptif dapat dibagi menjadi pendekatan berbasis model dan pendekatan bebas model. Manajemen termal berbasis model mempelajari model suhu dari sistem komputasi. Survei tentang teknik pemodelan termal disediakan di Bagian 6.3. Jika suhu yang diprediksi melebihi ambang batas maka akan diambil tindakan untuk mencegah hot spot. Alih-alih mempelajari model suhu, manajemen termal adaptif bebas model mengadaptasi tindakan kontrol secara langsung berdasarkan umpan balik yang dikumpulkan dari sistem. Kedua teknik memantau sistem dan mengekstrak informasi dari dinamika sistem terkini untuk

mengembangkan model baik untuk prediksi suhu atau untuk pengambilan keputusan. Selanjutnya, kami membahas pekerjaan terbaru dalam manajemen termal adaptif berbasis model dan bebas model.

Pengelolaan Termal Adaptif Berbasis Model

Manajemen termal berbasis model juga dapat disebut sebagai manajemen termal prediktif. Teknik ini membuat keputusan kontrol berdasarkan dinamika sistem yang diproyeksikan. Selama prediksinya akurat, keadaan darurat termal dapat dihindari dengan melakukan tindakan yang tepat sebelumnya. Perbedaan utama di antara metode ini terletak pada model prediksi suhu yang diadopsi dan tindakan DTM. Pembahasan rinci tentang prediksi suhu disediakan di bagian 6.3. Pada bagian ini kami fokus pada aktuasi berdasarkan model prediktif.

Dua kenop yang umum digunakan oleh pengontrol DTM adalah migrasi beban kerja yang sadar suhu (atau alokasi beban kerja) dan penskalaan tegangan dan frekuensi dinamis (DVFS). Tujuan utama pengontrol DTM adalah untuk mencapai beban kerja yang terdistribusi secara merata baik secara spasial maupun temporal untuk membatasi kenaikan suhu. Kami fokus pada beberapa migrasi/alokasi beban kerja prediktif yang baru-baru ini diusulkan dan teknik berbasis DVFS dalam bab ini.

Migrasi Tugas Sadar Suhu Prediktif

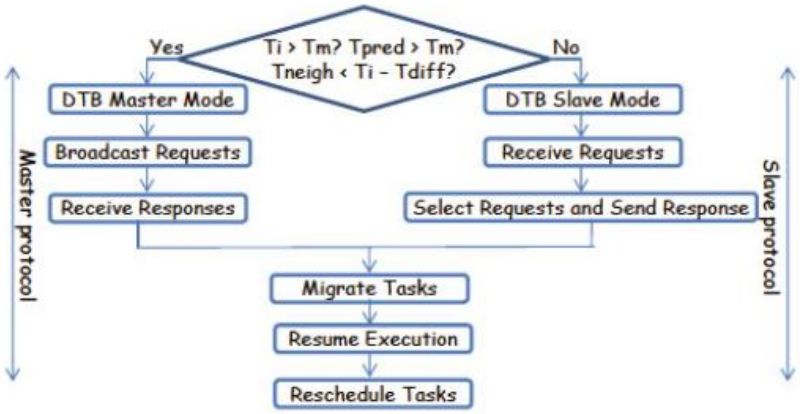
Dalam prosesor chip tunggal multi-core, suhu inti tidak hanya ditentukan oleh konsumsi daya program peranti lunak yang berjalan di atasnya, tetapi juga pada kemampuan pembuangan panasnya dan suhu tetangganya. Konsep dasar dari migrasi tugas yang sadar suhu adalah untuk secara dinamis memindahkan proses "panas" ke inti dengan pembuangan panas yang unggul dan/atau inti yang dikelilingi oleh tetangga yang lebih dingin sambil memindahkan proses "dingin" ke inti dengan pembuangan panas dan/atau yang lebih rendah. inti dikelilingi oleh tetangga yang lebih panas. Sebagian besar metode migrasi tugas sadar termal prediktif mengasumsikan bahwa konsumsi daya tugas dalam beban kerja tertentu dapat diperoleh baik dengan pembuatan profil atau estimasi. Semuanya memerlukan model prediksi suhu yang memperkirakan baik suhu keadaan tunak atau suhu di waktu mendatang.

Berdasarkan prediksi suhu, kebijakan migrasi tugas sadar termal yang disebut Predictive Dynamic Thermal Management (PDTM). Saat tugas yang berjalan di prosesor diproyeksikan melebihi ambang batas suhu, maka akan dipindahkan ke prosesor yang diprediksi paling keren di masa mendatang. Kebijakan tersebut dievaluasi pada prosesor Intel Quad-core. Suhu prosesor dapat dibaca dari sensor termal digital yang tertanam di dalam inti. Aplikasi yang berjalan pada sistem adalah libquantum, perlbench, bzip2, hmmer dari SPEC 2006 Benchmarks. Model prediksi mencapai akurasi yang sangat tinggi dan kesalahan prediksi suhu hanya 1,6%. Dibandingkan dengan skema manajemen termal yang ada, kebijakan migrasi tugas yang diusulkan mengurangi suhu rata-rata sebesar 7% dan suhu puncak sebesar 3 ◦C.

Penelitian Coskun mengusulkan kebijakan Penyeimbangan Termal Proaktif (PTB). Mirip dengan kebijakan PDTM, kebijakan ini memindahkan tugas dari inti prosesor yang diperkirakan lebih panas ke inti yang diperkirakan lebih dingin. Perbedaannya adalah PDTM memindahkan tugas yang sedang berjalan sementara PTB memberikan prioritas untuk memindahkan tugas dalam antrean tunggu. Hal ini memungkinkan inti panas memiliki periode dengan jumlah utas yang lebih sedikit dan menjadi dingin setelah tugas saat ini selesai, dan karenanya menghindari titik panas. Karena mekanisme migrasi tugas menunggu dalam ready queue telah diterapkan di penjadwal OS untuk tujuan penyeimbangan beban, teknik ini tidak memperkenalkan overhead implementasi tambahan. Hasil eksperimen pada model prosesor berbasis UltraSPARC T1 menunjukkan bahwa PTB mengurangi kejadian hot spot, gradien spasial, dan siklus termal masing-masing sebesar 60%, 80%, dan 75% dibandingkan dengan manajemen termal reaktif, di mana keputusan migrasi beban kerja dibuat setelah inti mencapai ambang tertentu. Selain itu, PTB hanya menimbulkan biaya kinerja kurang dari 2% sehubungan dengan kebijakan penjadwalan default untuk penyeimbangan muatan, sebagaimana diukur pada prosesor kehidupan nyata.

Baik PTB dan PDTM adalah pendekatan terpusat. Ini berarti bahwa teknik ini memerlukan pengontrol yang memantau distribusi suhu dan beban kerja seluruh chip dan membuat keputusan global tentang alokasi sumber daya. Pendekatan terpusat seperti itu mungkin tidak dapat diskalakan dengan baik ke sejumlah besar core. Seiring bertambahnya jumlah elemen pemrosesan, kompleksitas penyelesaian masalah manajemen sumber daya tumbuh secara super-linear. Selain itu, pemantauan terpusat dan kerangka perintah menimbulkan overhead yang besar, karena komunikasi antara pengontrol pusat dan inti akan meningkat secara eksponensial.

Kerangka kerja untuk manajemen termal terdistribusi di mana profil termal yang seimbang dapat dicapai dengan pelambatan termal serta migrasi tugas sadar termal di antara inti yang berdekatan. Framework memiliki agen berbiaya rendah yang berada di setiap inti. Agen mengamati beban kerja dan suhu prosesor lokal saat berkomunikasi dan bertukar tugas dengan tetangga terdekatnya. Tujuan dari migrasi tugas adalah untuk mendistribusikan tugas ke prosesor berdasarkan kemampuan pembuangan panasnya dan juga memastikan bahwa setiap prosesor memiliki perpaduan tugas daya tinggi dan daya rendah yang seimbang. Penulis mengacu pada teknik yang diusulkan sebagai migrasi penyeimbangan termal terdistribusi (DTB-M) karena bertujuan untuk menyeimbangkan beban kerja dan suhu prosesor secara bersamaan.



Gambar 6.8. Alur teknik Distributed Thermal Balancing-Migration (DTB-M).

Kebijakan DTB-M pada dasarnya dapat dibagi menjadi 3 fase: pemeriksaan dan prediksi suhu, pertukaran informasi, dan migrasi tugas. Gambar 6.8 menunjukkan flowchart eksekusi DTB-M di core ke-i. Agen DTB-M awalnya netral. Ini akan memasuki mode master jika salah satu dari tiga skenario benar: (1) Suhu lokal mencapai ambang batas Tm (dalam hal ini, DTB pertama-tama akan mematikan prosesor untuk mendinginkannya sebelum memasuki mode master), (2) prediksi suhu puncak masa depan melebihi ambang batas, (3) perbedaan suhu inti lokal dan inti tetangga melebihi ambang penyeimbang termal. Jika tidak, itu akan memasuki mode budak. Agen master DTB mengeluarkan permintaan migrasi tugas ke tetangga terdekatnya yang merupakan budak DTB.

Kebijakan DTB-M memiliki dua komponen. Keduanya memiliki kompleksitas kuadrat sehubungan dengan jumlah tugas dalam antrian tugas lokal. Komponen pertama adalah kebijakan migrasi berbasis suhu kondisi mapan (SSTM). Ini mempertimbangkan perilaku termal jangka panjang dari tugas, dan mendistribusikan tugas ke core berdasarkan kemampuan pembuangan panasnya yang berbeda. Komponen kedua DTB-M adalah kebijakan migrasi berdasarkan prediksi suhu (TPM), yang memprediksi suhu puncak dari kombinasi tugas yang berbeda saat membuat keputusan migrasi. Ini memastikan bahwa setiap inti menjalankan campuran seimbang tugas daya tinggi dan daya rendah tanpa darurat termal.

Kedua komponen ini saling melengkapi satu sama lain karena SSTM mempertimbangkan efek termal rata-rata jangka panjang dan TPM mempertimbangkan variasi temporal jangka pendek. Hasil percobaan menunjukkan bahwa DTB-M memiliki frekuensi titik panas 66,79% lebih rendah dan kinerja 40,21% lebih tinggi dibandingkan dengan teknik yang ada. Selain itu, DTB-M juga memiliki biaya migrasi yang jauh lebih rendah. Jumlah migrasi berkurang 33,84% sementara jarak migrasi keseluruhan berkurang 70,7% karena prosesor hanya berkomunikasi dan bertukar tugas dengan tetangga terdekatnya.

DVFS Prediktif untuk Manajemen Termal

Penelitian Cochran memanfaatkan karakteristik internal program untuk memprediksi suhu masa depan. Kita tahu hari ini bahwa pelaksanaan program

biasanya mencakup beberapa fase yang berbeda. Fase program adalah tahap eksekusi di mana beban kerja menunjukkan karakteristik daya, suhu, atau kinerja yang hampir identik. Oleh karena itu, selama kita dapat mengidentifikasi fase program selama run time dan membangun hubungan antara suhu operasi dan fase program, kita dapat memprediksi suhu masa depan dari sebuah aplikasi.

Karakteristik run time dari suatu program dapat dicirikan oleh kejadian arsitektural seperti instruksi per siklus (IPC), kecepatan akses memori, kecepatan cache miss dan instruksi *floating point* (FP) yang dieksekusi, dll. Setelah memilih kejadian yang paling relevan, langkah selanjutnya adalah mengelompokkan fase program yang serupa bersama-sama karena perilaku yang serupa akan menghasilkan suhu operasi yang serupa. Proses ini dilakukan pada kumpulan data pelatihan dengan titik data yang diekstraksi dari sekumpulan beban kerja yang representatif. Tujuan dari proses pengelompokan adalah untuk mengklasifikasikan *n* titik data ke *k* cluster sehingga jumlah jarak kuadrat antara setiap titik data pelatihan ke pusat clusternya diminimalkan. Kemudian model linier digunakan untuk merepresentasikan hubungan antara suhu dan fase beban kerja. Model dilatih menggunakan pemasangan kuadrat terkecil. Pada setiap interval waktu reguler, informasi penghitung kejadian dikumpulkan dan model suhu dievaluasi untuk setiap pengaturan frekuensi yang tersedia. Frekuensi tertinggi yang tidak menyebabkan pelanggaran termal dipilih untuk dijalankan pada interval berikutnya. Hasil eksperimen menunjukkan bahwa, dibandingkan dengan kebijakan manajemen termal berbasis DVFS reaktif, algoritma ini sangat mengurangi pelanggaran termal sekaligus mengurangi runtime sebesar 7,6%.

Sementara semua teknik yang disebutkan di atas bertujuan untuk mengurangi titik panas suhu sebagai tujuan utama, untuk melakukan DVFS untuk minimalisasi energi dan manajemen suhu. Sebuah energi dipisahkan dan pengontrol termal diperkenalkan dalam pekerjaan ini. Berdasarkan perkiraan jumlah siklus jam per instruksi (CPI) dan kendala kinerja, pengontrol energi menghitung lintasan frekuensi optimal *(f EC(t))* yang meminimalkan energi sekaligus memenuhi persyaratan kinerja. Pengontrol termal memilih frekuensi clock yang memiliki deviasi terkecil dari *f EC(t)* sambil mempertahankan suhu inti di bawah ambang batas yang diberikan. Pengontrol termal memprediksi suhu di masa depan menggunakan model ARX (AutoRegressive eXogenous), yang berasal dari rutinitas kalibrasi sendiri.

Pengelolaan Termal Adaptif Bebas Model

Sementara DTM berbasis model memilih tindakan berdasarkan suhu yang diprediksi, manajemen termal bebas model mempelajari tindakan secara langsung. Pembelajaran penguatan (RL) untuk masalah DTM untuk aplikasi multimedia. Mereka memodelkan masalah DTM sebagai proses kontrol stokastik dan mengadopsi algoritma RL untuk menemukan kebijakan optimal selama runtime.

Mereka memodelkan pengontrol DTM prosesor sebagai agen pembelajaran dan menganggap sistem lainnya sebagai lingkungan. Setelah agen pembelajaran melakukan tindakan, ia mengamati lingkungan dan memperkirakan hadiah atau

hukuman yang ditimbulkan oleh tindakan ini. Agen belajar dari pengalaman ini dan mencoba untuk meningkatkan keputusan masa depannya untuk memaksimalkan hadiah melalui model pembelajaran. Model pembelajaran yang diusulkan memiliki overhead waktu berjalan yang sangat kecil. Itu hanya menimbulkan beberapa pencarian tabel dan beberapa operasi aritmatika sederhana. Dibandingkan dengan kebijakan tanpa manajemen termal, kebijakan pembelajaran ini mengurangi pelanggaran termal sebesar 34,27% sambil mempertahankan waktu pengoperasian yang serupa.

Penelitian Coskun mengusulkan algoritma pembelajaran berbasis ahli switching untuk manajemen termal. Teknik pembelajaran mereka memanfaatkan serangkaian kebijakan manajemen, termasuk manajemen daya dinamis untuk mematikan inti menganggur (DPM), DVFS, migrasi thread, dan penyeimbangan beban. Kebijakan ini disebut sebagai ahli. Di atas para ahli, ada spesialis yang merupakan kebijakan tingkat yang lebih tinggi yang menentukan ahli mana yang akan digunakan pada interval berikutnya. Setiap spesialis ditugaskan dengan bobot yang merupakan fungsi dari kinerja dan suhu. Pada setiap interval waktu, hanya pakar yang memilih pakar aktif yang bobotnya diperbarui. Spesialis yang memiliki bobot tertinggi dipilih dan kebijakan pakar yang sesuai dijalankan. Melalui pemutakhiran bobot untuk para spesialis, teknik ini menjamin untuk menyatu dengan kebijakan yang paling sesuai dengan dinamika beban kerja saat ini. Hasil percobaan menunjukkan bahwa teknik pembelajaran ini mampu meningkatkan perilaku termal sistem, mengurangi energi, dan meningkatkan kinerja secara signifikan dibandingkan dengan kebijakan penjadwalan default di OS.

Manajemen Termal Berbasis Pembelajaran di Tingkat Mesin dan Server

Teknik DTM berbasis pembelajaran tidak terbatas pada mikroprosesor dan dapat diterapkan pada tingkat yang lebih tinggi, seperti pusat data. Untuk pusat data, masalah utamanya adalah menjaga suhu sasis server di bawah ambang batas sembari mengurangi daya komputasi dan konsumsi daya sistem pendingin, misalnya kipas pendingin atau Computer Room Air Condition (CRAC), sebanyak mungkin mungkin. Algoritma penempatan beban kerja untuk pusat data besar. Algoritma mereka bergantung pada prediktor permintaan beban kerja eksponensial untuk meramalkan permintaan pendapatan di masa depan ke pusat data. Alasan di balik model prediksi ini adalah bahwa beban kerja pusat data biasanya menunjukkan pola berulang dengan periode dalam urutan jam, hari, minggu, dan seterusnya. Keakuratan model prediksi tinggi: permintaan masuk yang diprediksi berada dalam 5% dari permintaan masuk yang sebenarnya. Teknik tersebut menggunakan Integer Linear Programming (ILP) untuk mengalokasikan beban kerja dan mengaktifkan atau menonaktifkan server tertentu untuk meminimalkan total energi pusat data. Hasil percobaan menunjukkan bahwa algoritma mereka mencapai penghematan energi yang konsisten dibandingkan dengan algoritma rakus selama simulasi 24 jam.

6.5 KESIMPULAN

Thermal Management adalah masalah desain yang menarik dan menantang untuk prosesor generasi berikutnya. Gradien termal meningkat dengan setiap generasi teknologi baru, memaksa desainer untuk berpikir di luar ranah tradisional. Setiap fase desain yang dibahas dalam bab ini saling bergantung dan mendukung manajemen termal dinamis. Pada prosesor generasi mendatang, mekanisme prediksi dan respons dapat memperoleh manfaat lebih jauh dari penggunaan algoritma berbasis evolusioner untuk manajemen termal. Sebagai contoh, pengontrol run-time untuk mengelola kebijakan termal yang berbeda menggunakan model non-linear yang kompleks dan konstanta waktu yang terlibat dalam mencapai solusi yang akurat adalah besar. Desainer dapat melihat algoritma lean yang memberikan solusi numerik yang lebih cepat dan sederhana. Algoritme semacam itu dapat bermanfaat dengan mengabstraksi koefisien desain dari hierarki yang berbeda, termasuk level termal, elektrik, dan sistem. Tren penting lainnya dalam prosesor generasi masa depan adalah penggunaan integrasi 3D, dengan standar JEDEC yang ditetapkan baru-baru ini. Manajemen termal dalam sistem ini membutuhkan kesadaran akan perilaku fisik tumpukan yang berbeda, mekanisme pendinginan terkait, mekanisme kontrol suhu yang menghubungkan seluruh tumpukan, dan algoritme yang secara optimal mendistribusikan atau mengalokasikan sumber daya berdasarkan gradien termal.

BAB 7
KOMUNIKASI NIRKABEL ON-CHIP UNTUK SISTEM MULTI-CORE

Paradigma Network-on-Chip telah muncul sebagai metodologi yang memungkinkan untuk mengintegrasikan sejumlah besar core fungsional pada satu die. Namun, jaringan on-chip multi-hop berbasis interkoneksi logam menghasilkan latensi tinggi dan pemborosan energi dalam transfer data. Untuk mengatasi masalah ini beberapa teknologi interkoneksi yang muncul telah diusulkan. Arsitektur NoC nirkabel terbukti mengungguli rekan-rekan kabel dengan beberapa urutan besarnya disipasi energi sambil mencapai kecepatan transfer data yang lebih tinggi. Namun, keandalan tautan nirkabel bersama dengan interkoneksi NoC logam diketahui menjadi perhatian utama dalam node teknologi masa depan. Kode kontrol kesalahan yang kuat berdasarkan kode produk dapat meningkatkan ketahanan saluran nirkabel sehingga membuat NoC secara keseluruhan lebih andal. Mekanisme kontrol kesalahan terpadu untuk meningkatkan ketahanan terhadap kesalahan transien dari tautan nirkabel serta tautan kabel disajikan. Bab ini menampilkan manfaat kinerja yang dapat dicapai dari arsitektur NoC nirkabel sambil tetap mempertahankan ketangguhan yang dapat diterima terhadap kesalahan sementara.

7.1 PENDAHULUAN

Desain sistem terintegrasi multi-core di luar era CMOS saat ini akan menghadirkan keuntungan dan tantangan yang belum pernah terjadi sebelumnya, yang pertama terkait dengan kepadatan perangkat yang sangat tinggi dan yang terakhir terkait masalah disipasi daya yang melonjak. Menurut International Technology Roadmap for Semiconductors (ITRS) pada tahun 2007, kontribusi interkoneksi terhadap disipasi daya chip diharapkan meningkat dari 51% pada generasi teknologi 0,13µm hingga 80% dalam periode lima tahun ke depan. Ini dengan jelas menunjukkan tantangan desain masa depan yang terkait dengan penskalaan tradisional interkoneksi logam konvensional dan inovasi material. Untuk meningkatkan kinerja chip multi-core berbasis interkoneksi logam konvensional, beberapa teknologi interkoneksi yang sangat berbeda saat ini sedang dieksplorasi seperti integrasi 3D, interkoneksi fotonik dan RF multi-band atau interkoneksi nirkabel.

Semua teknologi baru ini telah diprediksi mampu memungkinkan desain multi-core, yang meningkatkan kecepatan dan disipasi daya dalam transfer data secara signifikan. Desain arsitektur yang inovatif ditambah dengan teknologi interkoneksi yang muncul ini mencapai kinerja tinggi dalam transfer data on-chip sambil tetap membuang energi yang jauh lebih sedikit dibandingkan dengan NoC berbasis interkoneksi logam tradisional. Namun, paradigma interkoneksi alternatif ini sedang dalam tahap formatif dan perlu mengatasi tantangan signifikan yang berkaitan dengan keandalan. Error Control Coding (ECC) telah muncul sebagai solusi yang layak untuk meningkatkan keandalan link komunikasi on-chip. Telah ditunjukkan

bahwa ECC yang dirancang khusus meningkatkan kehandalan serta menurunkan energi disipasi link NoC tradisional.

Dalam bab ini kami menyajikan metodologi yang diilhami alam untuk merancang arsitektur NoC yang efisien dengan tautan nirkabel on-chip (WiNoC) dan mengatasi masalah keandalan dalam NoC nirkabel hibrid dengan kerangka kerja ECC terpadu dengan skema berbeda untuk tautan kabel dan nirkabel. Tautan nirkabel di NoC hibrid menghadapi tingkat kesalahan yang lebih tinggi daripada kabel biasa dan karenanya skema ECC yang lebih kuat diperlukan untuk memulihkan keandalan pada tautan tersebut. Dalam bab ini kami menentukan tingkat kesalahan tipikal dari tautan nirkabel on-chip dan menyajikan mekanisme ECC yang sesuai untuk meningkatkan keandalannya. Terlihat bahwa dengan menggunakan skema ECC terpadu yang diusulkan, dimungkinkan untuk mencapai disipasi energi yang sangat rendah dan lebar pita yang tinggi dalam NoC nirkabel sambil tetap mempertahankan keandalan yang serupa dengan mitra kabel biasa.

7.2 PEKERJAAN TERKAIT

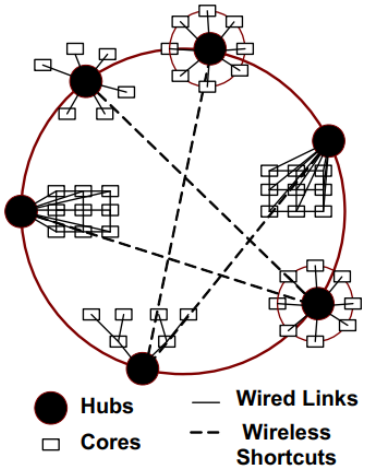
NoC konvensional menggunakan komunikasi packet switched multi-hop. Untuk meningkatkan kinerja. Terlihat bahwa dengan menggunakan jalur ekspres virtual untuk menghubungkan inti yang jauh dalam jaringan, adalah mungkin untuk menghindari overhead router pada node perantara, dan dengan demikian sangat meningkatkan kinerja NoC. NoC telah terbukti berkinerja lebih baik dengan memasukkan tautan kabel jarak jauh mengikuti prinsip grafik dunia kecil.

Prinsip-prinsip desain NoC tiga dimensi dan NoC fotonik diuraikan dalam berbagai publikasi terbaru. Diperkirakan bahwa NoC 3D dan fotonik akan menghilangkan daya yang jauh lebih sedikit daripada mitra elektroniknya. Alternatif lain untuk daya rendah adalah NoC dengan interkoneksi RF multi-band.

Baru-baru ini, desain NoC nirkabel berdasarkan teknologi CMOS Ultra Wideband (UWB). merancang jaringan komunikasi nirkabel on-chip dengan antena miniatur dan transceiver sederhana yang beroperasi pada kisaran sub-THz 100-500 GHz telah ditunjukkan.

Jika frekuensi transmisi dapat ditingkatkan ke kisaran THz maka ukuran antena yang sesuai akan berkurang, menempati jauh lebih sedikit ruang chip. Salah satu kemungkinan adalah dengan menggunakan antena nano berdasarkan CNT yang beroperasi di rentang frekuensi THz. Konsekuensinya, membangun jaringan interkoneksi nirkabel on-chip menggunakan frekuensi THz untuk komunikasi antar-inti menjadi layak. Semua teknologi interkoneksi yang muncul ini terbukti meningkatkan kinerja dan disipasi daya NoC. Namun karena fakta bahwa teknologi ini masih dalam tahap formatif, fabrikasi dan integrasinya dengan proses CMOS standar tidak dapat diandalkan. Selain itu, teknik tingkat sistem untuk mempertahankan perolehan kinerja di hadapan teknologi yang secara inheren tidak dapat diandalkan ini belum mendapat banyak perhatian dari para peneliti. Oleh karena itu, ada kebutuhan untuk mengeksplorasi arsitektur NoC yang akan mempertahankan keuntungan dari teknologi yang muncul tersebut bahkan di hadapan ketidakandalan yang melekat.

Secara khusus kami menyajikan desain NoC nirkabel menggunakan tautan nirkabel THz dengan antena berbasis CNT yang akan memungkinkan peningkatan kinerja dan disipasi daya yang tinggi meskipun tingkat kegagalan tautan nirkabel tinggi.



Gambar 7.1. Arsitektur NoC hybrid (nirkabel/kabel) konseptual dengan subnet heterogen

Transmisi sinyal di node teknologi saat ini dan masa depan menghadirkan tantangan yang belum pernah terjadi sebelumnya karena beberapa peristiwa. Dalam teknologi Ultra-Deep Submicron (UDSM) crosstalk coupling antara kabel yang berdekatan serta beberapa mekanisme kesalahan sementara seperti ground bounce dan partikel alfa menghasilkan peningkatan kemungkinan kesalahan multi-bit. Skema ECC telah dieksplorasi dalam konteks tautan NoC untuk meningkatkan keandalan transmisi data dan mengurangi pemborosan energi.

Dalam bab ini, pertama-tama kami akan menyajikan metodologi yang dapat digunakan untuk merancang NoC nirkabel di bagian 7.3 dan kemudian bagaimana ECC dapat digunakan untuk meningkatkan keandalan WiNoC di bagian 7.4.

7.3 ARSITEKTUR NOC NIRKABEL

Dalam NoC berkabel umum, inti tersemat konstituen berkomunikasi melalui beberapa sakelar dan tautan berkabel. Komunikasi multi-hop ini menghasilkan transfer data dengan disipasi energi dan latensi yang tinggi. Untuk mengatasi masalah ini kami mengusulkan tautan nirkabel bandwidth tinggi jarak jauh antara inti yang jauh di dalam chip. Pada subbagian berikut kami akan menjelaskan desain arsitektur yang dapat diskalakan untuk WiNoC dengan berbagai ukuran sistem.

Topologi

Teori jaringan kompleks modern memberi kita metode yang kuat untuk menganalisis topologi jaringan dan propertinya. Antara reguler, jaringan mesh yang saling terhubung secara lokal dan topologi Erdos-Renyi yang benar-benar acak, ada kelas graf lain, seperti graf dunia kecil dan graf bebas skala. Jaringan dengan properti dunia kecil memiliki panjang jalur rata-rata yang sangat pendek, yang biasanya diukur sebagai jumlah lompatan antara setiap pasangan node. Rata-rata panjang lintasan terpendek dari graf dunia kecil dibatasi oleh

polinomial dalam log(N), di mana N adalah jumlah simpul, yang membuatnya sangat menarik untuk komunikasi yang efisien dengan sumber daya minimal. Fitur grafik dunia kecil ini membuatnya sangat menarik untuk membuat WiNoC yang dapat diskalakan.

Sebagian besar jaringan yang kompleks, seperti jejaring sosial, Internet, serta bagian otak tertentu menunjukkan properti dunia kecil. Topologi dunia kecil dapat dibangun dari jaringan yang terhubung secara lokal dengan menghubungkan kembali koneksi secara acak ke node lain mana pun, yang menciptakan jalan pintas dalam jaringan. Tautan jarak jauh acak antara node ini juga dapat dibuat mengikuti distribusi probabilitas tergantung pada jarak yang memisahkan node. Telah ditunjukkan bahwa "pintasan" seperti itu dalam NoC dapat secara signifikan meningkatkan kinerja dibandingkan dengan jaringan mirip jala yang saling terhubung secara lokal dengan sumber daya yang lebih sedikit daripada sistem yang terhubung sepenuhnya.

Tujuan kami di sini adalah menggunakan pendekatan "dunia kecil" untuk membangun NoC yang sangat efisien berdasarkan tautan kabel dan nirkabel. Jadi, untuk tujuan kami, pertama-tama kami membagi seluruh sistem menjadi beberapa kelompok kecil dari inti tetangga dan menyebut jaringan yang lebih kecil ini subnet. Karena subnet adalah jaringan yang lebih kecil, komunikasi intra-subnet akan memiliki panjang jalur rata-rata yang lebih pendek daripada satu NoC yang menjangkau seluruh sistem. Gambar 7.1 memperlihatkan arsitektur WiNoC dengan beberapa subnet dengan topologi mesh, star, ring dan tree. Subnet memiliki sakelar dan tautan NoC seperti pada NoC standar. Inti terhubung ke hub yang terletak di pusat melalui tautan langsung dan hub dari semua subnet terhubung dalam jaringan tingkat 2 yang membentuk jaringan hierarkis. Tingkat atas hierarki ini dirancang untuk memiliki karakteristik grafik dunia kecil.

Karena terbatasnya jumlah link nirkabel yang mungkin, seperti yang dibahas dalam subbagian selanjutnya, hub tetangga dihubungkan oleh link kabel tradisional yang membentuk cincin dan beberapa link nirkabel didistribusikan antara hub yang dipisahkan oleh jarak yang relatif jauh. Mengurangi komunikasi kabel multi-hop jarak jauh sangat penting untuk mencapai manfaat penuh dari jaringan nirkabel on-chip untuk sistem multi-core. Karena tautan awalnya dibuat secara probabilistik, kinerja jaringan mungkin tidak optimal. Oleh karena itu, setelah penempatan awal tautan nirkabel, jaringan selanjutnya dioptimalkan kinerjanya dengan menggunakan Simulated Annealing (SA). Distribusi probabilitas tertentu dan heuristik yang diikuti dalam membangun tautan jaringan dijelaskan pada sub-bagian berikutnya. Kunci pendekatan kami adalah membangun topologi jaringan keseluruhan yang optimal di bawah batasan sumber daya yang diberikan, yaitu, sejumlah tautan nirkabel. Gambar 7.1 menunjukkan kemungkinan topologi interkoneksi. Alih-alih cincin yang digunakan dalam contoh ini, hub dapat dihubungkan dalam arsitektur interkoneksi lainnya yang memungkinkan. Ukuran dan jumlah subnet dipilih sedemikian rupa sehingga baik subnet maupun level atas hierarki tidak menjadi terlalu besar. Ini karena jika salah satu level hierarki menjadi terlalu besar maka akan menyebabkan hambatan kinerja dengan membatasi throughput data di level tersebut. Namun, karena arsitektur dari dua tingkat dapat berbeda menyebabkan karakteristik lalu lintas mereka berbeda satu sama lain.

Kami mengusulkan arsitektur NoC berkabel/nirkabel hybrid. Hub saling terhubung melalui tautan nirkabel dan kabel sementara subnet hanya terhubung dengan kabel. Hub dengan tautan nirkabel dilengkapi dengan stasiun pangkalan nirkabel (WB) yang mengirim dan menerima paket data melalui saluran nirkabel. Ketika sebuah paket perlu dikirim ke inti dalam subnet yang berbeda, ia bergerak dari sumber ke hub masing-masing dan mencapai hub subnet tujuan melalui jaringan dunia kecil yang terdiri dari tautan nirkabel dan kabel, di mana ia kemudian dialihkan. ke inti tujuan akhir. Untuk transmisi data antar-subnet dan intra-subnet, perutean lubang cacing diadopsi. Paket data dipecah menjadi bagian-bagian yang lebih kecil yang disebut flow control unit atau flit. Header flit menyimpan informasi routing dan kontrol. Ini menetapkan jalur, dan muatan selanjutnya atau body flits mengikuti jalur itu.

Penyisipan dan Pengoptimalan Tautan Nirkabel

Seperti disebutkan di atas, keseluruhan infrastruktur interkoneksi WiNoC dibentuk dengan menghubungkan inti di subjaringan satu sama lain dan ke hub pusat melalui kabel logam tradisional. Hub kemudian dihubungkan dengan kabel dan tautan nirkabel sedemikian rupa sehingga jaringan tingkat 2 memiliki properti dunia kecil. Penempatan tautan nirkabel antara sepasang hub sumber dan tujuan tertentu penting karena ini bertanggung jawab untuk membangun interkoneksi berkecepatan tinggi dan berenergi rendah pada jaringan, yang pada akhirnya akan menghasilkan peningkatan kinerja. Awalnya tautan ditempatkan secara probabilistik; yaitu, antara setiap pasang hub sumber dan tujuan, masing-masing i dan j, probabilitas Pij untuk memiliki tautan nirkabel sebanding dengan jarak yang diukur dalam jumlah lompatan sepanjang cincin, hij, seperti yang ditunjukkan pada Persamaan (7.1).

$$P_{ij} = \frac{h_{ij}}{\sum_{i,j} h_{ij}}. \qquad (7.1)$$

Probabilitas dinormalisasi sedemikian rupa sehingga jumlahnya sama dengan satu. Distribusi seperti itu dipilih karena dengan adanya tautan nirkabel, jarak antara pasangan menjadi satu lompatan dan karenanya mengurangi jarak asli antara hub yang berkomunikasi melalui cincin. Bergantung pada jumlah tautan nirkabel yang tersedia, mereka disisipkan di antara pasangan hub yang dipilih secara acak, yang dipilih mengikuti distribusi probabilitas yang disebutkan di atas.

Setelah jaringan diinisialisasi, pengoptimalan melalui heuristik SA dilakukan. Karena arsitektur subnet tidak bergantung pada jaringan tingkat atas, pengoptimalan hanya dapat dilakukan pada jaringan hub tingkat atas dan karenanya sub-jaringan dapat dipisahkan dari langkah ini. Langkah optimisasi diperlukan karena inisialisasi acak mungkin tidak menghasilkan topologi jaringan yang optimal. SA menawarkan pendekatan yang sederhana, mapan, dan dapat diskalakan untuk proses pengoptimalan sebagai lawan dari pencarian brute force.

Jika ada N hub dalam jaringan dan n tautan nirkabel untuk didistribusikan, ukuran ruang pencarian S diberikan oleh:

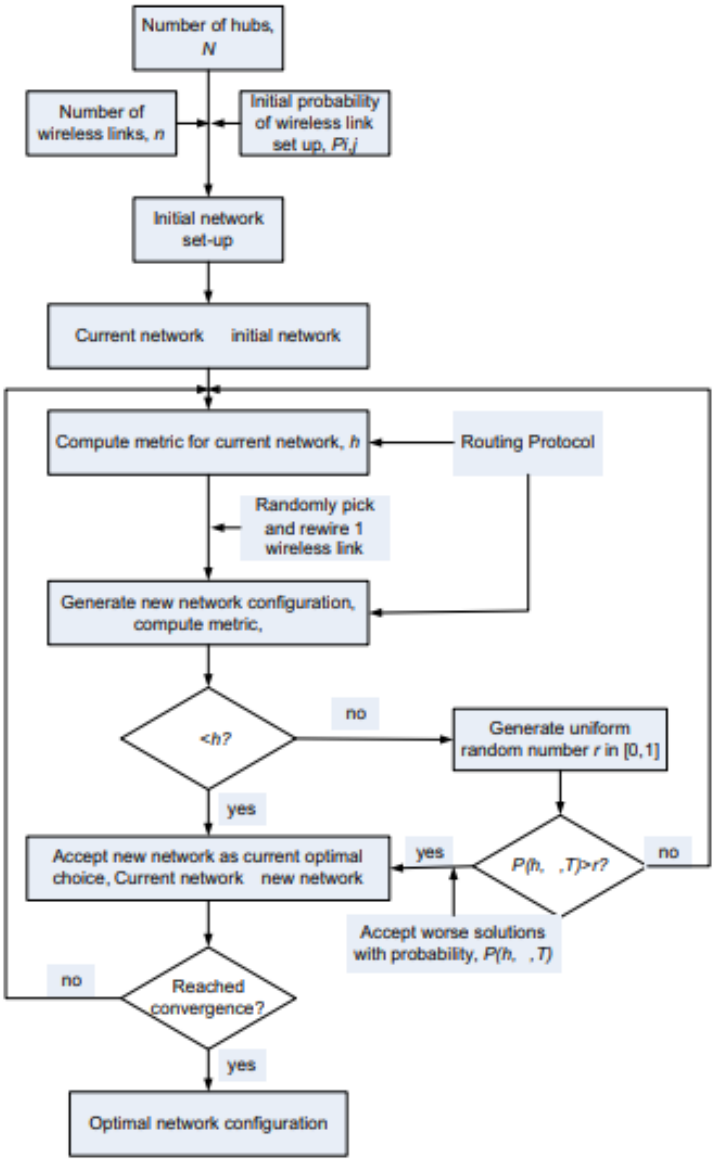
$$|S| = \binom{\binom{N}{2} - N}{n} \qquad (7.2)$$

Dengan demikian, dengan meningkatnya N, semakin sulit untuk menemukan solusi terbaik dengan pencarian lengkap. Untuk melakukan SA, metrik telah ditetapkan, yang terkait

erat dengan konektivitas jaringan. Metrik yang akan dioptimalkan adalah jarak rata-rata, diukur dalam jumlah lompatan, antara semua hub sumber dan tujuan.

Untuk menghitung metrik ini, jarak terpendek antara semua pasangan hub dihitung mengikuti strategi perutean. Dalam setiap iterasi proses SA, sebuah jaringan baru dibuat dengan memasang ulang secara acak tautan nirkabel di jaringan saat ini. Metrik untuk jaringan baru ini dihitung dan dibandingkan dengan metrik jaringan saat ini. Jaringan baru selalu dipilih sebagai solusi optimal saat ini jika metriknya lebih rendah. Namun, meskipun metriknya lebih tinggi, kami memilih jaringan baru secara probabilistik. Ini mengurangi kemungkinan terjebak dalam optimum lokal, yang bisa terjadi jika proses SA tidak pernah memilih solusi yang lebih buruk. Probabilitas eksponensial yang ditunjukkan pada Persamaan (7.3) digunakan untuk menentukan apakah solusi yang lebih buruk dipilih sebagai optimal saat ini atau tidak:

$$P(h,h',T) = exp[(h-h')/T]. \qquad (7.3)$$



Gambar 7.2. Diagram alir untuk pengoptimalan arsitektur WiNoC berbasis anil yang disimulasikan

Metrik pengoptimalan untuk jaringan saat ini dan baru masing-masing adalah h dan h∗. T adalah parameter temperatur, yang menurun dengan jumlah iterasi optimasi menurut jadwal anil. Di sini kita telah menggunakan penjadwalan Cauchy, dimana temperatur bervariasi berbanding terbalik dengan jumlah iterasi. Algoritma yang digunakan untuk mengoptimalkan jaringan ditunjukkan pada Gambar 7.2.

Di sini kita asumsikan distribusi lalu lintas spasial yang seragam di mana sebuah paket yang berasal dari inti mana pun memiliki kemungkinan yang sama untuk memiliki inti lain pada die sebagai tujuannya. Namun, dengan jenis distribusi lalu lintas spasial lainnya, di mana beban jaringan dilokalkan dalam kelompok yang berbeda, metrik untuk pengoptimalan harus diubah untuk memperhitungkan pola lalu lintas yang tidak seragam seperti yang dibahas nanti di bagian selanjutnya.

Komponen penting dalam desain WiNoC adalah antena on-chip untuk sambungan nirkabel. Pada bagian selanjutnya kami menjelaskan berbagai pilihan antena on-chip alternatif dan kelebihan dan kekurangannya.

Antena On-Chip

Antena on-chip yang sesuai diperlukan untuk membangun tautan nirkabel untuk WiNoC. Dalam kinerja antena on-chip terintegrasi silikon untuk komunikasi intra- dan antar-chip. Mereka terutama menggunakan antena zig-zag logam yang beroperasi pada kisaran puluhan GHz. Desain antena ultra wideband (UWB) untuk komunikasi antar dan intra-chip. Antena khusus ini digunakan dalam desain NoC nirkabel yang disebutkan sebelumnya di bagian 7.2. Antena yang disebutkan di atas pada prinsipnya beroperasi dalam kisaran gelombang milimeter (puluhan GHz) dan akibatnya ukurannya berada di urutan beberapa milimeter.

Jika frekuensi transmisi dapat ditingkatkan ke THz/rentang optik maka ukuran antena yang sesuai akan berkurang, menempati chip real estate yang jauh lebih sedikit. Karakteristik antena logam yang beroperasi di wilayah optik dan inframerah dekat dari spektrum hingga 750 THz telah dipelajari.

Karakteristik antena tabung nano karbon (CNT) dalam rentang frekuensi THz/optik juga telah diselidiki baik secara teoritis maupun eksperimental. Bundel CNT diperkirakan akan meningkatkan kinerja modul antena hingga 40dB dalam efisiensi radiasi dan memberikan sifat arah yang sangat baik dalam pola medan jauh. Selain itu antena ini dapat mencapai bandwidth sekitar 500 GHz, sedangkan antena yang beroperasi pada rentang gelombang milimeter mencapai bandwidth puluhan GHz. Dengan demikian, antena yang beroperasi pada rentang frekuensi THz/optik dapat mendukung kecepatan data yang jauh lebih tinggi. CNT memiliki banyak karakteristik yang membuatnya cocok sebagai elemen antena on-chip untuk frekuensi optik. Mengingat panjang gelombang ratusan nanometer hingga beberapa mikrometer, ada kebutuhan akan struktur antena satu dimensi untuk transmisi dan penerimaan yang efisien. Dengan diameter beberapa nanometer dan panjang hingga beberapa milimeter, CNT adalah kandidat yang sempurna.

Struktur tipis seperti itu hampir tidak mungkin dicapai dengan teknik mikrofabrikasi tradisional untuk logam. Struktur CNT yang hampir bebas cacat tidak mengalami kehilangan daya karena kekasaran permukaan dan ketidaksempurnaan tepi yang ditemukan pada antena

metalik tradisional. Dalam CNT, transpor elektron balistik mengarah ke konduktansi kuantum, menghasilkan pengurangan kerugian resistif, yang memungkinkan kerapatan arus yang sangat tinggi di CNT, yaitu 4-5 kali lipat lebih tinggi daripada tembaga. Ini memungkinkan daya pancar tinggi dari antena nanotube, yang sangat penting untuk komunikasi jarak jauh. Dengan menyorotkan sumber laser eksternal pada CNT, karakteristik radiasi antena tabung nano karbon multi-dinding (MWCNT) diamati dalam kesesuaian kuantitatif yang sangat baik dengan teori antena radio tradisional, meskipun pada frekuensi yang jauh lebih tinggi dari ratusan THz. Menggunakan berbagai panjang elemen antena yang sesuai dengan kelipatan berbeda dari panjang gelombang laser eksternal, pola hamburan dan radiasi terbukti ditingkatkan. Antena nanotube seperti itu adalah kandidat yang baik untuk membuat tautan komunikasi nirkabel on-chip dan selanjutnya dipertimbangkan dalam bab ini.

Deposisi uap kimia (CVD) adalah metode tradisional untuk menumbuhkan nanotube di lokasi tertentu dengan menggunakan pulau katalis berpola litograf. Penerapan medan listrik selama pertumbuhan atau arah aliran gas selama CVD dapat membantu menyelaraskan nanotube. Namun, CVD suhu tinggi berpotensi merusak beberapa lapisan CMOS yang sudah ada sebelumnya. Untuk meringankan ini, pemanas lokal dalam proses fabrikasi CMOS untuk mengaktifkan CVD nanotube lokal tanpa memaparkan seluruh chip ke suhu tinggi digunakan.

Seperti disebutkan di atas, NoC dibagi menjadi beberapa subnet. Oleh karena itu, WB di subnet perlu dilengkapi dengan antena pengirim dan penerima, yang akan terhubung menggunakan sumber laser eksternal. Sumber laser dapat ditempatkan di luar chip atau terikat pada cetakan silikon. Karenanya disipasi daya mereka tidak berkontribusi pada kerapatan daya chip. Persyaratan menggunakan sumber eksternal untuk menggairahkan antena dapat dihilangkan jika fenomena electroluminescence dari CNT digunakan untuk merancang sumber radiasi dipol terpolarisasi linier. Tetapi penyelidikan lebih lanjut diperlukan untuk menetapkan perangkat seperti itu sebagai transceiver yang berhasil untuk komunikasi nirkabel on-chip.

Untuk mencapai komunikasi line of sight antara WB menggunakan antena CNT pada frekuensi optik, bahan kemasan chip harus diangkat dari permukaan substrat untuk menciptakan ruang hampa untuk transmisi gelombang EM frekuensi tinggi. Teknik untuk membuat kemasan vakum seperti itu sudah digunakan untuk aplikasi MEMS, dan dapat diadopsi untuk membuat pembuatan komunikasi line of sight antara antena CNT yang layak. Dalam teori antena klasik diketahui bahwa daya yang diterima menurun berbanding terbalik dengan daya 4 pemisahan antara sumber dan tujuan akibat pantulan tanah di luar jarak tertentu. Pemisahan ambang ini, r0 antara antena sumber dan tujuan dengan asumsi permukaan yang memantulkan sempurna, diberikan oleh Persamaan (7.4):

$$r_0 = \frac{2 \cdot \Pi \cdot H^2}{\lambda}. \qquad (7.4)$$

Di sini H adalah ketinggian antena di atas permukaan pemantul dan λ adalah panjang gelombang pembawa. Jadi, jika elemen antena berada pada jarak H dari permukaan reflektif seperti dinding kemasan dan bagian atas substrat cetakan, daya yang diterima menurun secara terbalik dengan kuadrat jarak sampai r0. Dengan demikian H dapat disesuaikan untuk membuat pemisahan maksimum yang mungkin lebih kecil dari ambang pemisah r0 untuk

frekuensi radiasi tertentu yang digunakan. Mempertimbangkan rentang frekuensi optik antena CNT, tergantung pada pemisahan antara pasangan sumber dan tujuan dalam satu chip, elevasi yang diperlukan hanya beberapa puluh mikron.

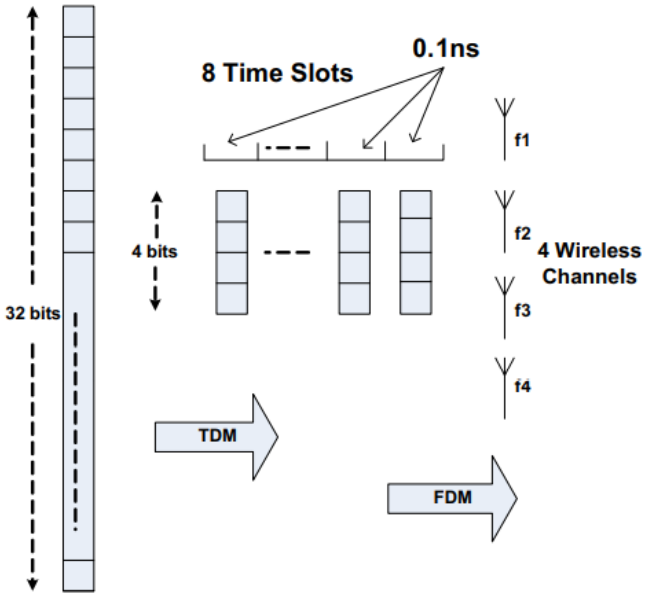
Routing dan Protokol Komunikasi

Dalam WiNoC yang diusulkan, perutean data intra-subnet bergantung pada topologi subnet. Misalnya, jika inti dalam subnet terhubung dalam arsitektur mesh, maka perutean data dalam subnet mengikuti perutean urutan dimensi (e-cube). Data antar-subnet dirutekan melalui hub, sepanjang jalur terpendek antara subnet sumber dan tujuan dalam hal jumlah tautan yang dilalui. Hub di semua subnet dilengkapi dengan blok pra-perutean untuk menentukan jalur ini melalui pencarian di semua jalur potensial antara hub subnet sumber dan tujuan. Dalam pekerjaan saat ini, jalur yang hanya melibatkan satu tautan nirkabel dan tidak ada atau sejumlah tautan kabel apa pun pada ring dipertimbangkan. Semua jalur seperti itu serta jalur kabel lengkap pada ring dibandingkan dan jalur dengan jumlah minimum traversal tautan dipilih untuk transfer data. Untuk paket data yang membutuhkan perutean antar-subnet, perhitungan ini dilakukan hanya sekali untuk header yang melayang di hub subnet asal. Flit header harus memiliki bidang yang berisi alamat hub perantara dengan WB yang akan digunakan di jalur. Hanya informasi ini yang cukup karena header mengikuti jalur wireline default di sepanjang ring ke hub tersebut dengan WB dari sumbernya, yang juga merupakan jalur terpendek di sepanjang ring. Karena setiap WB memiliki satu tujuan unik, header mencapai tujuan tersebut dan kemudian dirutekan lagi melalui jalur wireline ke hub tujuan akhirnya menggunakan perutean cincin normal. Flit lainnya mengikuti header seperti yang biasanya terjadi pada perutean lubang cacing.

Pendekatan perutean alternatif adalah untuk menghindari evaluasi satu kali terpendek jalur di hub sumber asli dan mengadopsi mekanisme perutean terdistribusi. Dalam skenario ini, jalur ditentukan pada setiap node dengan memeriksa keberadaan link nirkabel pada node tersebut, yang jika diambil akan mempersingkat panjang jalur ke tujuan akhir. Jika tautan nirkabel ini tidak ada atau mempersingkat jalur dibandingkan dengan jalur wireline dari node tersebut, maka mekanisme perutean default di sepanjang ring akan diikuti ke node berikutnya. Mekanisme ini melakukan pemeriksaan pada setiap node dengan menghitung dan membandingkan panjang jalur dengan menggunakan perutean wireline default atau tautan nirkabel. Perutean terpusat yang diadopsi melakukan semua pemeriksaan di hub sumber asli, yang mencakup semua tautan nirkabel dan jalur kabel dari sumber ke tujuan.

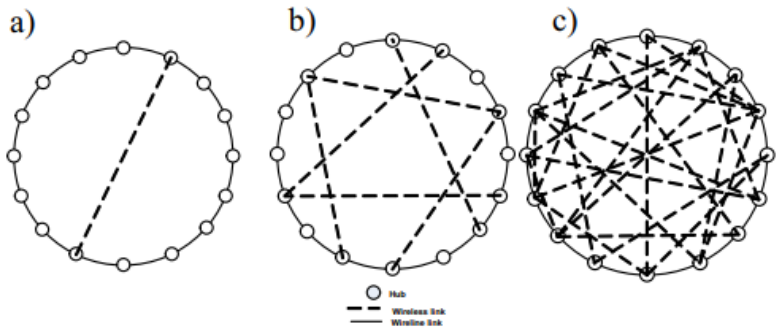
Dengan menggunakan sumber laser multiband untuk menggairahkan antena CNT, saluran frekuensi yang berbeda dapat ditetapkan ke pasangan subnet yang berkomunikasi. Ini akan membutuhkan penggunaan elemen antena yang disetel ke frekuensi yang berbeda untuk setiap pasangan, sehingga menciptakan bentuk multiplexing pembagian frekuensi (FDM). Hal ini dimungkinkan dengan menggunakan CNT dengan panjang yang berbeda, yang merupakan kelipatan dari panjang gelombang dari masing-masing frekuensi pembawa. Keuntungan arah yang tinggi dari antena ini, membantu dalam menciptakan saluran terarah antara pasangan sumber dan tujuan. 24 sumber laser gelombang kontinyu dari frekuensi yang berbeda digunakan. Dengan demikian, 24 frekuensi yang berbeda ini dapat ditetapkan ke beberapa tautan nirkabel di WiNoC sedemikian rupa sehingga saluran frekuensi tunggal digunakan

hanya sekali untuk menghindari interferensi sinyal pada frekuensi yang sama. Ini memungkinkan penggunaan saluran multi-band secara bersamaan melalui chip. Jumlah tautan nirkabel dalam jaringan dapat bervariasi dari 24 tautan, masing-masing dengan saluran frekuensi tunggal, hingga satu tautan dengan semua 24 saluran. Menetapkan beberapa saluran per tautan meningkatkan bandwidth tautan.



Gambar 7.3. Protokol komunikasi yang diadopsi untuk saluran nirkabel

Saat ini, modulator dan demodulator optik Mach-Zehnder terintegrasi silikon berkecepatan tinggi, yang mengubah sinyal listrik menjadi sinyal optik dan sebaliknya tersedia secara komersial. Modulator optik dapat memberikan kecepatan data 10 Gbps per saluran pada tautan ini. Di penerima, penguat derau rendah (LNA) dapat digunakan untuk meningkatkan kekuatan sinyal listrik yang diterima, yang kemudian akan disalurkan ke subnet tujuan. Kecepatan data ini diperkirakan akan meningkat berlipat ganda dengan penskalaan teknologi masa depan. Skema modulasi yang diadopsi adalah non-coherent on-off keying (OOK), dan oleh karena itu tidak memerlukan rangkaian pemulihan dan sinkronisasi jam yang rumit. Karena keterbatasan jumlah saluran frekuensi berbeda yang dapat dibuat melalui antena CNT, lebar flit di NoC umumnya lebih tinggi daripada jumlah saluran yang mungkin per link. Jadi, untuk mengirim seluruh flit melalui tautan nirkabel menggunakan sejumlah frekuensi berbeda yang terbatas, skema penyaluran yang tepat perlu diadopsi. Di sini kita asumsikan lebar layar 32 bit. Oleh karena itu, untuk mengirim seluruh flit menggunakan saluran frekuensi yang berbeda, time division multiplexing (TDM) diadopsi. Berbagai komponen saluran nirkabel yaitu, modulator elektro-optik, modulator/demodulator TDM, LNA dan router untuk merutekan data pada jaringan hub diimplementasikan sebagai bagian dari WB. Gambar 7.3 mengilustrasikan mekanisme komunikasi yang diadopsi untuk transfer data antar-subnet.



Gambar 7.4. Pengaturan tautan nirkabel yang optimal untuk (a) 1, (b) 6, dan (c) 24 tautan nirkabel di antara 16 hub. Perhatikan bahwa solusi simetris dengan kinerja yang sama dimungkinkan

Dalam contoh WiNoC ini, kami menggunakan tautan nirkabel dengan 4 saluran frekuensi. Dalam hal ini, satu flit dibagi menjadi 8 gigitan empat bit, dan setiap gigitan diberikan slot waktu 0,1ns, sesuai dengan laju bit 10 Gbps. Bit di setiap nibble ditransmisikan secara bersamaan melalui empat frekuensi pembawa yang berbeda.

Mekanisme perutean yang dibahas dalam bagian ini mudah diperluas untuk menggabungkan teknik pengalamatan lain seperti multicasting. Performa arsitektur NoC tradisional yang menggabungkan multicasting dan dapat juga digunakan untuk meningkatkan performa WiNoC yang dikembangkan. Sebagai contoh, mari kita perhatikan sebuah subnet dalam sistem 16-subnet (seperti pada Gambar 7.4), yang mencoba mengirim paket ke 3 subnet lain sedemikian rupa sehingga salah satunya berlawanan secara diagonal dengan subnet sumber dan dua lainnya aktif. kedua sisi itu. Dengan tidak adanya tautan nirkabel jarak jauh, menggunakan multicasting latensi beban nol untuk pengiriman satu flit adalah 9 siklus sedangkan tanpa multicasting, flit yang sama akan membutuhkan 11 siklus untuk dikirim ke tujuan masing-masing. Di sini komunikasi hanya terjadi di sepanjang ring. Namun, jika tautan nirkabel ada di sepanjang diagonal dari subnet sumber ke tujuan tengah, maka dengan multicasting, flit dapat ditransfer dalam 5 siklus jika ada 8 saluran berbeda dalam tautan tersebut. Empat siklus diperlukan untuk mentransfer flit 32-bit ke hub yang berseberangan secara diagonal melalui tautan nirkabel dan satu lompatan lagi di sepanjang ring ke tujuan akhir di kedua sisi. Efisiensi penggunaan multicasting bervariasi dengan jumlah saluran dalam tautan karena mengatur bandwidth tautan nirkabel.

7.4 EVALUASI KINERJA

Pada bagian ini kami menganalisis karakteristik arsitektur WiNoC yang diusulkan dan mempelajari tren kinerjanya dengan penskalaan ukuran sistem. Untuk analisis kami, kami telah mempertimbangkan tiga ukuran sistem yang berbeda, yaitu 128, 256, dan 512 inti pada cetakan berukuran 20mm 20mm. Kami mengamati hasil peningkatan ukuran sistem dengan meningkatkan jumlah subnet serta jumlah inti per subnet. Oleh karena itu, dalam satu skenario, kami menganggap jumlah inti per subnet tetap adalah 16 dan memvariasikan jumlah subnet antara 8, 16, dan 32. Dalam kasus lain, kami mempertahankan jumlah subnet tetap pada 16 dan bervariasi ukuran subnet dari 8 hingga 32 core. Konfigurasi sistem ini dipilih

berdasarkan eksperimen yang akan dijelaskan nanti di bagian selanjutnya. Pembentukan tautan nirkabel menggunakan anil simulasi, bagaimanapun, hanya bergantung pada jumlah hub pada tingkat ke-2 jaringan.

Pembentukan Tautan Nirkabel

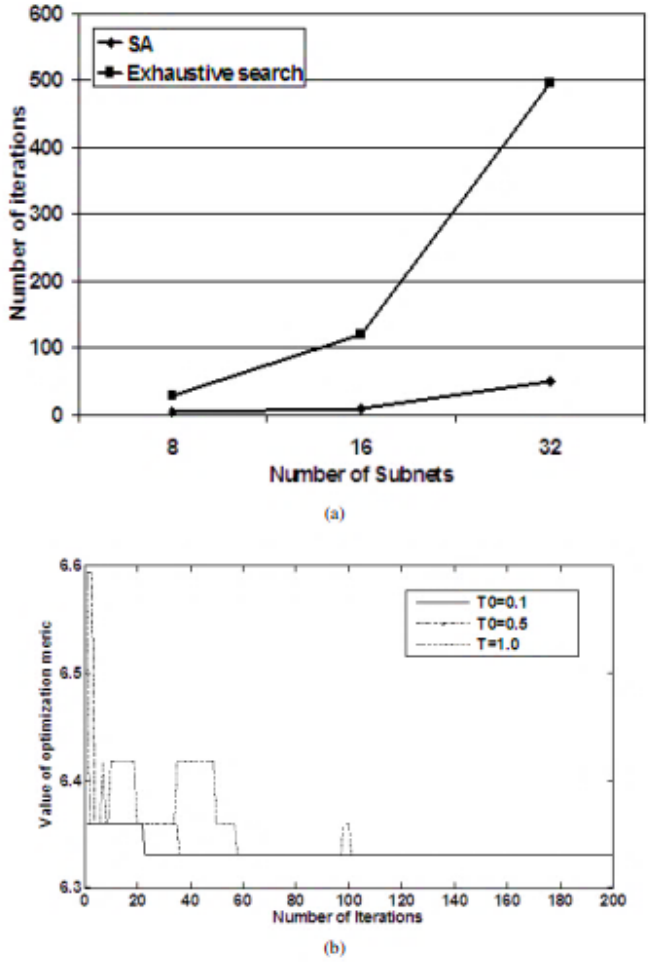
Awalnya hub terhubung dalam sebuah cincin melalui kabel normal dan sambungan nirkabel dibuat antara hub yang dipilih secara acak mengikuti distribusi probabilitas yang diberikan oleh Persamaan (7.1). Kami kemudian menggunakan anil simulasi untuk mencapai konfigurasi optimal dengan menemukan posisi tautan nirkabel yang meminimalkan jarak rata-rata antara semua pasangan sumber dan tujuan dalam jaringan. Gambar 7.4 menunjukkan lokasi 1, 6 dan 24 sambungan nirkabel dengan masing-masing 24, 4 dan 1 saluran dalam jaringan 16 hub. Kami mengikuti metodologi pengoptimalan yang sama untuk semua jaringan lainnya. Jarak rata-rata yang sesuai untuk jaringan yang dioptimalkan dengan ukuran sistem yang berbeda ditunjukkan pada Tabel 7.1.

Tabel 7.1. Jarak rata-rata untuk WiNoC yang dioptimalkan

No. of Sub-net (N)	Rata-rata jarak antar HOP		
	1 wireless link	24 wireless link	24 wireless link
8	1.7188	1.3125	1.1250 (12 Link)
16	3.2891	2.1875	1.5625
32	6.3301	3.8789	2.6309

*Dalam kasus 8 subnet, hanya 12 tautan nirkabel yang digunakan dengan 2 saluran per tautan.

Perlu dicatat bahwa penempatan link nirkabel tertentu untuk mendapatkan konfigurasi jaringan yang optimal tidaklah unik karena pertimbangan simetris dalam pengaturan kami, yaitu, ada beberapa konfigurasi dengan kinerja optimal yang sama. Untuk menetapkan kinerja algoritma SA yang digunakan, kami membandingkan metrik pengoptimalan yang dihasilkan dengan metrik yang diperoleh melalui pencarian menyeluruh untuk konfigurasi jaringan yang dioptimalkan untuk berbagai ukuran sistem. Algoritma SA menghasilkan konfigurasi jaringan dengan jumlah hop rata-rata total yang persis sama dengan yang dihasilkan oleh teknik pencarian lengkap untuk konfigurasi sistem yang dibahas dalam bab ini.



Gambar 7.5. (a) Jumlah iterasi yang diperlukan untuk mencapai solusi optimal dengan SA dan metode pencarian lengkap, (b) Konvergensi dengan suhu berbeda

Namun, konfigurasi WiNoC yang diperoleh dalam hal topologi tidak unik karena konfigurasi yang berbeda dapat memiliki jumlah hop rata-rata yang sama. Gambar 7.5(a) menunjukkan jumlah iterasi yang diperlukan untuk mencapai solusi optimal dengan SA dan algoritme pencarian lengkap. Jelas algoritma SA konvergen ke konfigurasi optimal jauh lebih cepat daripada teknik pencarian lengkap. Keuntungan ini akan meningkat untuk ukuran sistem yang lebih besar. Gambar 7.5(b) menunjukkan konvergensi metrik untuk nilai suhu awal yang berbeda untuk mengilustrasikan bahwa pendekatan SA konvergen dengan kuat ke nilai optimal dari rata-rata hopcount dengan variasi numerik pada suhu. Simulasi ini dilakukan untuk sistem dengan 32 subnet dengan 1 wireless link. Dengan nilai suhu awal yang lebih tinggi, dibutuhkan waktu lebih lama untuk menyatu. Secara alami, untuk nilai suhu awal yang cukup besar, metrik tidak menyatu. Di sisi lain, nilai suhu awal yang lebih rendah membuat sistem konvergen lebih cepat tetapi berisiko terjebak dalam optimum lokal. Dengan menggunakan konfigurasi jaringan yang dikembangkan dalam subbagian ini, kami sekarang akan mengevaluasi kinerja WiNoC berdasarkan metrik kinerja yang telah mapan.

Metrik Kinerja

Untuk mencirikan performa arsitektur WiNoC yang diusulkan, kami mempertimbangkan tiga parameter jaringan: latensi, throughput, dan disipasi energi. Latensi

mengacu pada jumlah siklus clock antara injeksi header pesan di node sumber dan penerimaan tail flit di tujuan. Throughput didefinisikan sebagai jumlah rata-rata flits yang berhasil diterima per inti tertanam per siklus clock. Disipasi energi per paket adalah energi rata-rata yang dihamburkan oleh satu paket ketika disalurkan dari node sumber ke node tujuan; subnet kabel dan saluran nirkabel berkontribusi untuk ini. Untuk subnet, sumber disipasi energi adalah kabel antar-saklar dan blok sakelar. Untuk saluran nirkabel, kontribusi utama berasal dari WB, yang meliputi antena, sirkuit transceiver dan modul komunikasi lainnya seperti blok TDM dan LNA. Disipasi energi per paket, Epkt, dapat dihitung menurut Persamaan (7.5) di bawah ini.

$$E_{pkt} = \frac{N_{intra-subnet} \cdot E_{subnet,hop} \cdot h_{subnet} + N_{inter-subnet} \cdot E_{sw} \cdot h_{sw}}{N_{intra-subnet} + N_{inter-subnet}} \qquad (7.5)$$

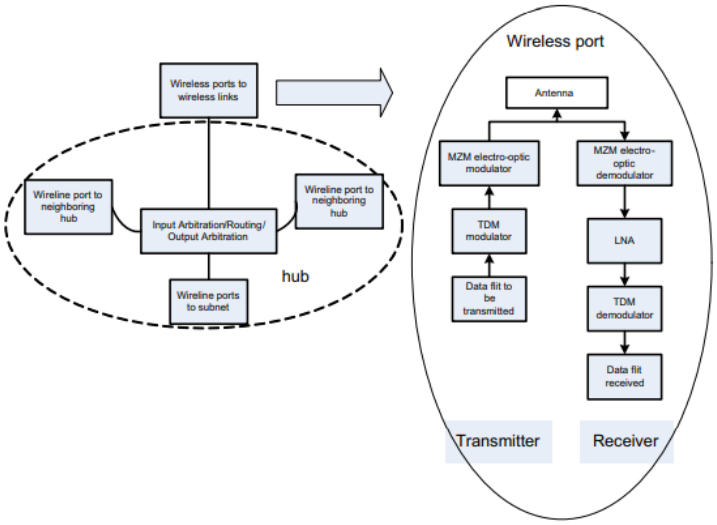
Dalam Persamaan (7.5), Nintrasubnet dan Nintersubnet adalah jumlah total paket yang dirutekan di dalam subnet dan di antara subnet. *Esubnet,hop* adalah energi yang dihamburkan oleh sebuah paket yang melintasi sebuah hop tunggal pada subnet berkabel termasuk tautan dan sakelar kabel, dan *Esw* adalah energi yang hilang oleh sebuah paket yang melintasi sebuah hop tunggal pada tingkat jaringan ke-2 dari *WiNoC*, yang mana memiliki sifat dunia kecil. *Esw* juga menyertakan disipasi energi di inti ke tautan hub. Dalam Persamaan (7.5), hsubnet dan *hsw* adalah rata-rata jumlah hop per paket di subnet dan jaringan dunia kecil.

Evaluasi Kinerja

Arsitektur jaringan yang dikembangkan sebelumnya di bagian ini disimulasikan menggunakan simulator siklus akurat yang memodelkan progres perpindahan data secara akurat per siklus jam yang memperhitungkan perpindahan yang mencapai tujuan dan juga yang dijatuhkan. Seratus ribu iterasi dilakukan untuk mencapai hasil yang stabil di setiap kasus, menghilangkan efek transien dalam beberapa ribu siklus pertama.

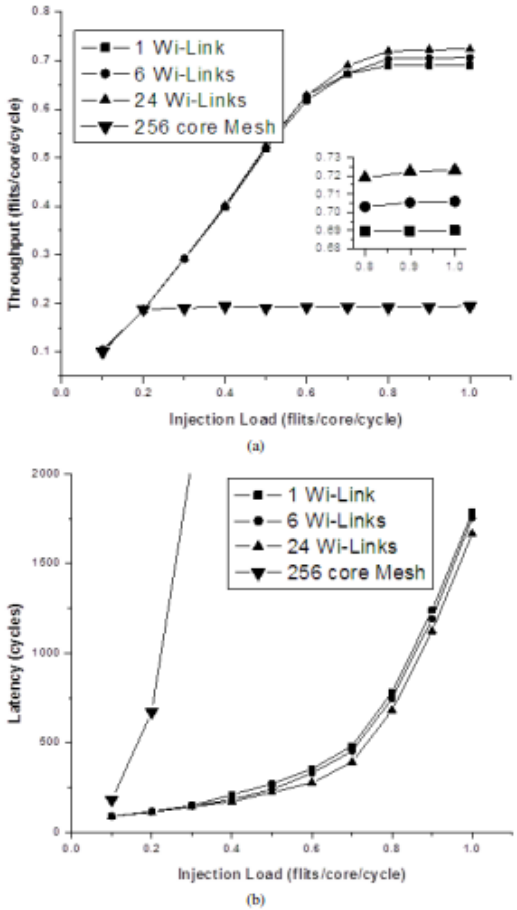
Arsitektur subnet mesh dipertimbangkan. Lebar semua tautan kabel dianggap sama dengan ukuran flit, yaitu 32 di sini. Arsitektur switch NoC tertentu, memiliki tiga tahap fungsional, yaitu, arbitrase input, routing/switch traversal, dan arbitrasi output. Port input dan output termasuk yang ada di link nirkabel memiliki empat saluran virtual per port, masing-masing memiliki kedalaman buffer 2 flits. Setiap paket terdiri dari 64 flits. Mirip dengan komunikasi intra-subnet, kami juga mengadopsi perutean lubang cacing di saluran nirkabel. Konsekuensinya, hub memiliki arsitektur yang mirip dengan sakelar NoC di subnet. Oleh karena itu, setiap port hub memiliki arbiter input dan output yang sama, dan jumlah saluran virtual yang sama dengan kedalaman buffer yang sama dengan subnet switch. Jumlah port di hub tergantung pada jumlah tautan yang terhubung dengannya. Port nirkabel dari WB diasumsikan dilengkapi dengan antena, modul TDM, dan modulator dan demodulator elektro-optik. Berbagai komponen WB ditunjukkan pada Gambar 7.6.

Hub yang hanya terdiri dari port ke tautan kabel juga disorot pada gambar untuk menekankan bahwa WB memiliki komponen tambahan dibandingkan dengan hub. Sakelar dan hub NoC pipelined, dan tautan kabel digerakkan dengan clock frekuensi 2,5 GHz.



Gambar 7.6. Komponen WB dengan beberapa port kabel dan nirkabel pada input dan output

Mengacu pada gaambar 7.7 menunjukkan plot throughput dan latensi sebagai fungsi dari beban injeksi untuk sistem dengan 256 core yang dibagi menjadi 16 subnet, masing-masing dengan 16 core. Keterlambatan yang ditimbulkan oleh tautan kabel dari inti ke hub untuk berbagai jumlah inti di subnet untuk ukuran sistem yang berbeda ditunjukkan pada Tabel 7.2.



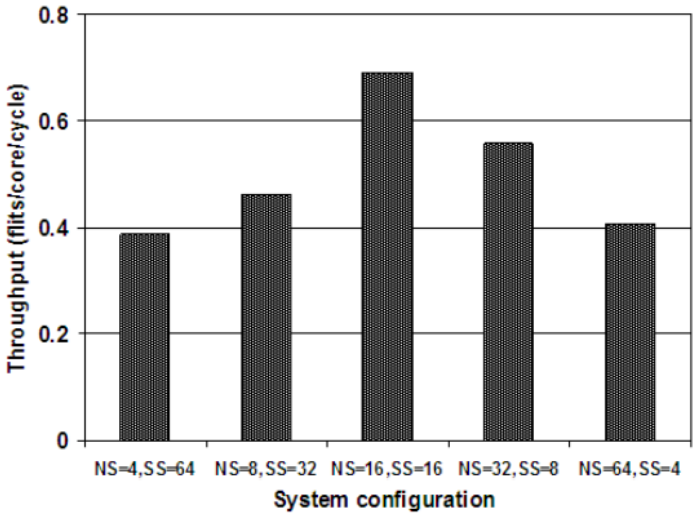
Gambar 7.7. (a) Throughput dan (b) latensi 256 WiNoC inti dengan jumlah tautan nirkabel yang berbeda

Tabel 7.2. Penundaan pada tautan kabel di WiNoCs

Sistem	*Ukuran Jumlah subnet*	*Ukuran subnet*	*Penundaan tautan hub inti (ps)*	*Penundaan tautan antar-hub (ps)*
128	8	16	96	181/86*
	16	8	60	86
256	16	16	60	86
	16	32	60	86
512	32	16	48	86/43*

*untuk 8 dan 32 subnet jarak antar-subnet berbeda sepanjang dua arah planar.

Penundaan kabel antar-hub untuk berbagai jumlah subnet juga ditampilkan. Seperti dapat dilihat penundaan ini semua kurang dari periode clock 400ps dan dapat dicatat bahwa panjang kedua inti-ke-hub dan link wireline antar-hub akan berkurang dengan peningkatan jumlah subnet sebagai masing-masing subnet menjadi lebih kecil di area dan subnet juga menjadi lebih dekat satu sama lain. Penundaan yang ditimbulkan oleh konversi sinyal elektro-optik dengan perangkat MZM adalah 20ps. Saat menghitung latensi sistem secara keseluruhan dan throughput WiNoC, penundaan komponen individual ini diperhitungkan. Topologi hierarki khusus ini dipilih karena memberikan kinerja sistem yang optimal. Gambar 7.8 menunjukkan throughput saturasi untuk cara alternatif membagi 256 inti WiNoC menjadi sejumlah subnet yang berbeda dengan satu tautan nirkabel. Seperti dapat dilihat dari plot, semua konfigurasi alternatif mencapai throughput saturasi yang lebih buruk. Kecenderungan yang sama diamati jika kita memvariasikan jumlah tautan nirkabel. Dengan menggunakan metode yang sama, pembagian hirarki yang sesuai yang mencapai kinerja terbaik ditentukan untuk semua ukuran sistem lainnya. Untuk ukuran sistem 128 dan 512, divisi hierarki yang dipertimbangkan di sini mencapai kinerja yang jauh lebih baik dibandingkan dengan divisi lain yang memungkinkan dengan jumlah subnet yang lebih rendah atau lebih tinggi.

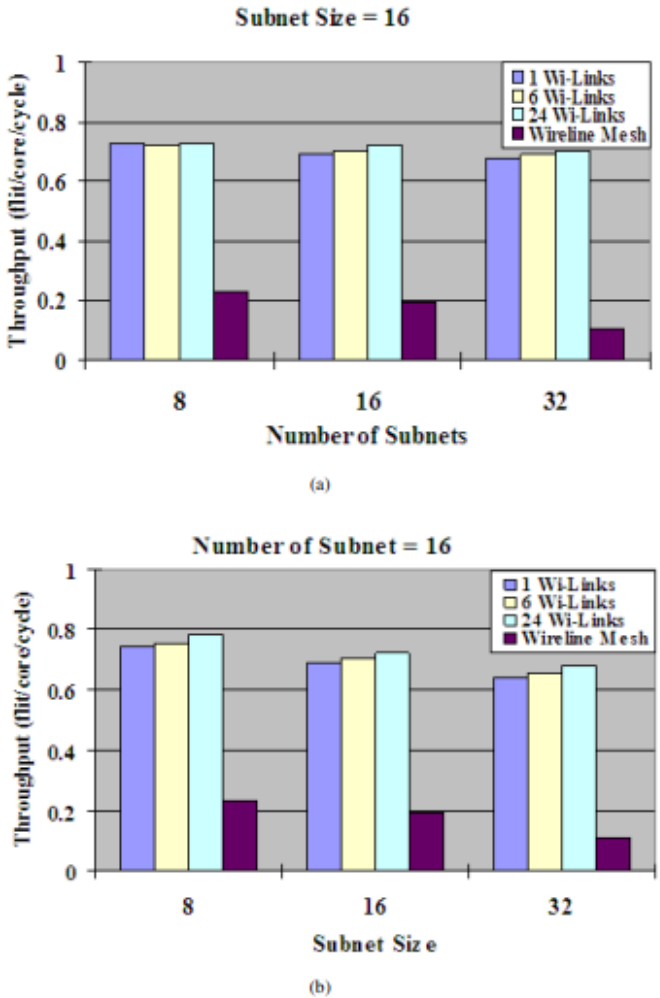


Gambar 7.8. Throughput 256 core WiNoC untuk berbagai konfigurasi hierarkis

Dengan memvariasikan jumlah saluran dalam tautan nirkabel, berbagai konfigurasi WiNoC dibuat. Kami telah mempertimbangkan WiNoC dengan 1, 6, dan 24 tautan nirkabel

dalam penelitian kami. Karena jumlah total frekuensi yang dipertimbangkan di sini adalah 24, jumlah saluran per tautan masing-masing adalah 24, 4 dan 1.

Seperti dapat dilihat dari Gambar 7.8, WiNoC dengan kemungkinan konfigurasi yang berbeda mengungguli arsitektur flat mesh monolitik berkabel tunggal. Dapat juga diamati bahwa dengan meningkatnya jumlah tautan nirkabel, throughput sedikit meningkat. Perlu dicatat bahwa meskipun peningkatan jumlah tautan memang meningkatkan jumlah tautan komunikasi nirkabel bersamaan, bandwidth pada setiap tautan berkurang karena jumlah total saluran ditetapkan oleh jumlah sumber laser off-chip. Ini menyebabkan total bandwidth di semua saluran nirkabel tetap sama. Satu-satunya perbedaan adalah tingkat distribusi di seluruh jaringan. Konsekuensinya, throughput jaringan hanya meningkat sedikit dengan meningkatnya jumlah tautan nirkabel. Namun, biaya perangkat keras meningkat dengan meningkatnya jumlah tautan. Dengan demikian, tergantung pada apakah permintaan pada kinerja sangat penting, perancang dapat memilih untuk menukar overhead area dengan menyebarkan jumlah maksimum tautan nirkabel yang mungkin. Namun, jika kendala pada overhead area benar-benar ketat maka seseorang dapat memilih untuk menggunakan hanya satu sambungan nirkabel dan akibatnya menyediakan lebih banyak bandwidth per sambungan dan hanya memiliki sedikit efek negatif pada kinerja.



Gambar 7.9. Throughput saturasi dengan (a) jumlah subnet yang bervariasi dan (b) ukuran masing-masing subnet

Untuk mengamati tren di antara berbagai konfigurasi WiNoC, kami melakukan analisis lebih lanjut. Gambar 7.9(a) menunjukkan throughput pada saturasi jaringan untuk berbagai ukuran sistem sambil menjaga ukuran subnet tetap untuk jumlah sambungan nirkabel yang berbeda. Gambar 7.9(b) menunjukkan variasi throughput pada saturasi untuk ukuran sistem yang berbeda untuk jumlah subnet yang tetap. Sebagai perbandingan, throughput pada saturasi jaringan untuk satu NoC wired mesh tradisional dari setiap ukuran sistem juga ditampilkan di kedua plot. Seperti pada Gambar 7.8 dapat dicatat dari Gambar 7.9 bahwa untuk WiNoC dengan ukuran tertentu, jumlah subnet dan ukuran subnet, throughput meningkat dengan peningkatan jumlah tautan nirkabel yang digunakan.

Seperti yang dapat diamati dari plot, throughput maksimum yang dapat dicapai dalam WiNoC menurun dengan meningkatnya ukuran sistem untuk kedua kasus. Namun, dengan meningkatkan jumlah subnet, penurunan throughput menjadi lebih kecil dibandingkan saat ukuran subnet ditingkatkan. Dengan meningkatkan ukuran subnet, kami meningkatkan kemacetan di subnet berkabel dan memuat di hub dan tidak sepenuhnya menggunakan kapasitas tautan nirkabel berkecepatan tinggi di tingkat atas jaringan. Ketika jumlah subnet bertambah, kemacetan lalu lintas di subnet tidak menjadi lebih buruk dan penempatan link nirkabel yang optimal membuat jaringan tingkat atas sangat efisien untuk transfer data. Pengaruh pada throughput dengan peningkatan ukuran sistem karena itu marjinal.

Untuk menentukan karakteristik disipasi energi dari WiNoC, pertama-tama kami memperkirakan energi yang dihamburkan oleh elemen antena. Gain arah antena MWCNT yang kami usulkan untuk digunakan sangat tinggi. Rasio daya yang dipancarkan terhadap daya insiden berada di sepanjang arah penguatan maksimum. Dengan asumsi saluran line-of-sight yang ideal lebih dari beberapa milimeter, daya yang ditransmisikan menurun dengan jarak mengikuti hukum kuadrat terbalik. Oleh karena itu daya PR yang diterima dapat dikaitkan dengan daya yang dipancarkan P_T sebagai:

$$P_R = \frac{G_T \cdot A_R}{4 \cdot \Pi \cdot R^2} P_T. \qquad (7.6)$$

Pada Persamaan (7.6), GT adalah penguatan antena pemancar, yang dapat diasumsikan -5dB. AR adalah luas antena penerima dan R adalah jarak antara pemancar dan penerima. Disipasi energi antena pemancar karena itu tergantung pada jangkauan komunikasi. Luas antena penerima dapat diketahui dengan menggunakan konfigurasi antena. Ini menggunakan MWCNT dengan diameter 200 nm dan panjang 7λ, dimana λ adalah panjang gelombang optik. Panjang 7λ dipilih karena terbukti menghasilkan gain terarah tertinggi, GT, pada pemancar. Pada salah satu setup, panjang gelombang laser yang digunakan adalah 543,5 nm, sehingga panjang antena sekitar 3,8 µm. Dengan menggunakan dimensi ini, luas antena penerima, AT dapat dihitung.

Lantai kebisingan LNA adalah -101 dBm. Mempertimbangkan demodulator MZM menyebabkan kerugian tambahan hingga 3dB selama lebar pita operasional, sensitivitas penerima menjadi kasus terburuk. Panjang tautan nirkabel terpanjang yang dipertimbangkan di antara semua konfigurasi WiNoC adalah 23 mm. Untuk panjang dan sensitivitas penerima ini, diperlukan daya transmisi 1,3 mW. Mempertimbangkan disipasi energi pada antena pemancar dan penerima, dan komponen sirkuit pemancar dan penerima seperti MZM, blok

TDM dan LNA, disipasi energi dari sambungan nirkabel terpanjang pada chip adalah 0,33 pJ/bit. Disipasi energi dari tautan nirkabel, ELink diberikan sebagai

$$E_{Link} = \sum_{i=1}^{m} (E_{antenna,i} + E_{transceiver,i}), \quad (7.7)$$

di mana m adalah jumlah saluran frekuensi dalam tautan dan *Eantenna,i* dan *Etransceiver,i* adalah disipasi energi elemen antena dan sirkuit transceiver untuk frekuensi ke-i dalam tautan.

Switch dan hub jaringan disintesis dari desain level RTL menggunakan pustaka sel standar 65nm dari CMP, menggunakan *Synopsys Design Vision* dan mengasumsikan frekuensi clock 2,5 GHz. Satu set pola data yang besar dimasukkan ke dalam daftar jaringan tingkat gerbang dari sakelar dan hub jaringan, dan dengan menjalankan *S ynopsysTM Prime Power*, disipasi energinya diperoleh.

Disipasi energi dari tautan kabel tergantung pada panjangnya. Panjang kabel interswitch di subnet dapat ditemukan dengan menggunakan rumus

$$l_M = \frac{l_{edge}}{M-1}. \quad (7.8)$$

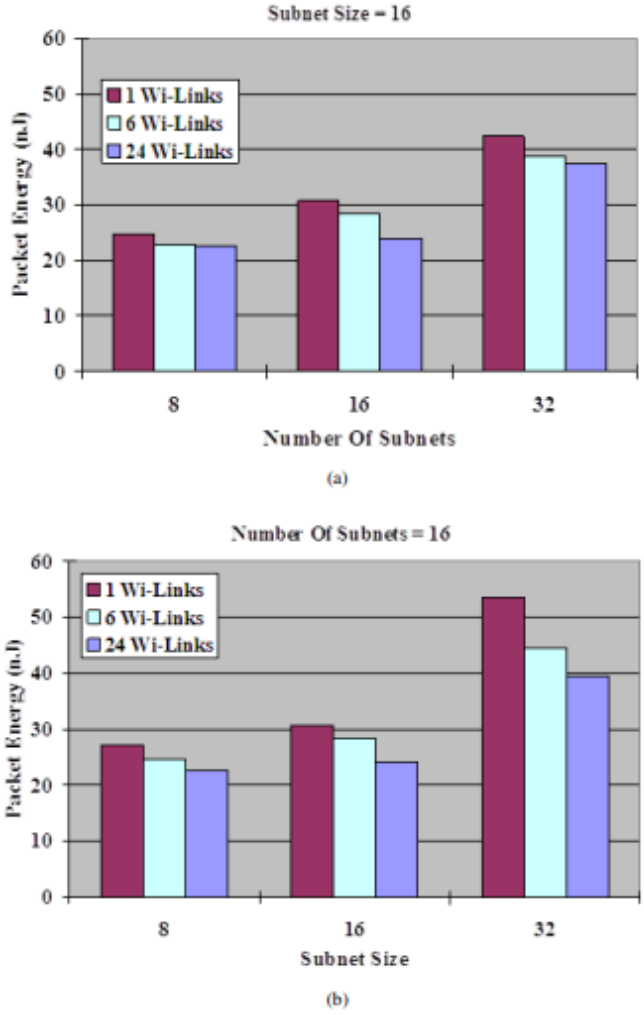
Di sini, M adalah jumlah inti di sepanjang tepi tertentu dari subnet dan langkan adalah panjang tepi tersebut. Ukuran die 20mm 20mm dipertimbangkan untuk semua ukuran sistem dalam simulasi kami. Panjang kabel antar-hub juga dihitung dengan cara yang sama karena ini diasumsikan dihubungkan dengan kabel yang sejajar dengan tepi cetakan hanya dalam dimensi persegi panjang. Oleh karena itu, untuk menghitung jarak antar-hub di sepanjang ring, sejajar dengan sisi tertentu dari die, Persamaan (7.8) dimodifikasi dengan mengubah M menjadi jumlah hub di sepanjang sisi tersebut dan langkan ke panjang sisi tersebut. Di setiap subnet, panjang tautan yang menghubungkan sakelar ke hub bergantung pada posisi sakelar dalam cetakan 20mm 20mm. Kapasitansi dari setiap tautan kabel, dan selanjutnya disipasi energinya, diperoleh melalui simulasi HSPICE dengan mempertimbangkan tata letak spesifik untuk subnet dan tingkat ke-2 jaringan cincin.

Gambar 10(a) dan 10(b) menunjukkan disipasi energi paket untuk setiap konfigurasi jaringan yang dipertimbangkan di sini. Energi paket untuk arsitektur flat wired mesh tidak ditampilkan karena lebih tinggi daripada WiNoC berdasarkan urutan besarnya, dan karenanya tidak dapat ditampilkan pada skala yang sama. Perbandingan dengan wired case ditunjukkan pada Tabel 7.3 pada sub-bagian berikutnya bersama dengan arsitektur hirarkis wired lainnya.

Tabel 7.3. Disipasi energi paket untuk arsitektur wired mesh datar, WiNoC dan hirarkis G-line NoC

Sistem	Ukuran Subnet	Jumlah Subnet Flat Mesh(nJ)		WiNoC (nJ)	NoC with G-Line (nJ)
128	16	8	1319	22.57	490.30
256	16	16	2936	24.02	734.50
512	16	32	4992	37.48	1012.81

Dari plot jelas bahwa disipasi energi paket meningkat dengan meningkatnya ukuran sistem. Namun, meningkatkan jumlah subnet berdampak lebih rendah pada energi paket rata-rata. Alasannya adalah throughput tidak banyak menurun dan latensi rata-rata per paket juga tidak berubah secara signifikan. Oleh karena itu, paket data menempati sumber daya jaringan untuk durasi yang lebih singkat, hanya menyebabkan sedikit peningkatan energi paket. Namun, dengan peningkatan ukuran subnet, throughput menurun secara nyata, begitu pula latensi. Dalam hal ini, energi paket meningkat secara signifikan karena setiap paket menempati sumber daya jaringan untuk jangka waktu yang lebih lama. Dengan peningkatan jumlah tautan nirkabel sambil mempertahankan jumlah subnet dan ukuran subnet konstan, energi paket berkurang.



Gambar 7.10. Disipasi energi paket dengan variasi (a) jumlah subnet dan (b) ukuran tiap subnet

Ini karena konektivitas jaringan yang lebih tinggi menghasilkan throughput yang lebih tinggi (atau latensi lebih rendah), yang berarti bahwa paket dialihkan lebih cepat, menggunakan sumber daya jaringan untuk waktu yang lebih sedikit, dan mengkonsumsi lebih sedikit energi selama transmisi. Karena subnet wireline menimbulkan kemacetan utama seperti yang dibuktikan oleh tren di plot Gambar 7.9 dan 7.10 ukurannya harus dioptimalkan. Dengan kata lain, subnet yang lebih kecil menyiratkan kinerja yang lebih baik dan energi paket

yang lebih rendah. Oleh karena itu, sebagai perancang, seseorang harus menargetkan untuk membatasi ukuran subnet selama ukuran tingkat atas jaringan tidak berdampak negatif pada kinerja sistem secara keseluruhan. Solusi optimal eksak juga bergantung pada arsitektur tingkat atas jaringan, yang tidak perlu dibatasi pada topologi ring yang dipilih dalam bab ini sebagai contoh.

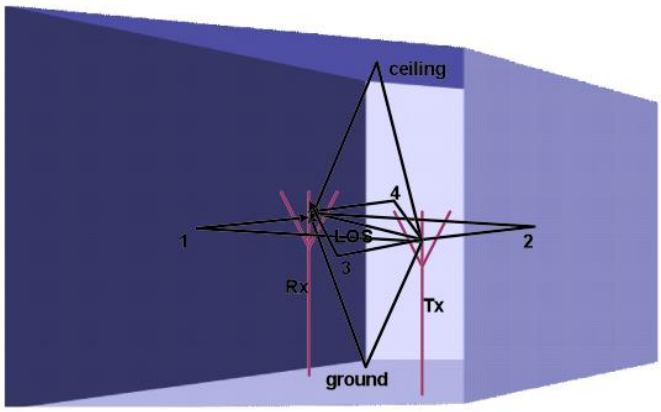
Strategi perutean terpusat yang diadopsi dibandingkan dengan perutean terdistribus untuk WiNoC berukuran 256 inti yang dibagi menjadi 16 subnet dengan masing-masing 16 inti. Dua puluh empat sambungan nirkabel dikerahkan dalam topologi yang sama persis untuk kedua kasus. Dengan routing terdistribusi throughput adalah 0,67 flits/inti/siklus sedangkan dengan perutean terpusat adalah 0,72 flits/inti/siklus seperti yang telah disebutkan pada Gambar 7.7, yang mana 7,5% lebih tinggi. Perutean terpusat menghasilkan throughput yang lebih baik karena menemukan jalur terpendek sedangkan perutean terdistribusi menggunakan jalur yang tidak optimal dalam beberapa kasus. Oleh karena itu, routing terdistribusi memiliki throughput yang lebih rendah. Perutean terdistribusi menghilangkan energi paket 31,2 nJ dibandingkan dengan 30,8 nJ dengan perutean terpusat. Hal ini karena rata-rata perhitungan panjang jalur dengan routing terdistribusi lebih banyak per paket, karena perhitungan ini terjadi pada setiap WB perantara. Namun, dengan perutean terpusat, setiap hub memiliki overhead perangkat keras tambahan untuk menghitung jalur terpendek dengan membandingkan semua jalur menggunakan tautan nirkabel.

7.5 KEANDALAN DALAM WINOC

Penskalaan yang agresif dalam node teknologi nanometer menghasilkan perangkat yang secara inheren tidak dapat diandalkan atau rawan cacat. Kinerja tautan nirkabel di WiNoC bergantung pada antena CNT. Seperti perangkat nano lainnya, antena CNT diharapkan memiliki tingkat cacat produksi yang lebih tinggi, ketidakpastian operasional, dan variabilitas proses. Error Control Coding (ECC) telah diusulkan untuk mengurangi masalah keandalan yang melekat dalam komunikasi on-chip. CNT yang rawan cacat dapat menyebabkan tingkat kesalahan acak yang relatif tinggi dan menyebabkan kesalahan multi-bit atau kesalahan burst. Oleh karena itu skema pengkodean harus kuat terhadap kesalahan acak dan meledak. Sebagai langkah pertama, pada subbagian berikutnya kami menyajikan model saluran nirkabel on-chip dan mengevaluasi tingkat kesalahan bit yang sesuai. Pada subbagian terakhir kami mengusulkan dan mengevaluasi skema ECC untuk tautan nirkabel on-chip dan untuk tautan kabel, untuk meningkatkan keandalan WiNoC.

Model Saluran Nirkabel

Dengan meninggikan bahan pengemas chip dari substrat untuk menciptakan ruang hampa untuk transmisi gelombang EM frekuensi tinggi, komunikasi LOS antara WB menggunakan antena CNT pada frekuensi optik dapat dicapai. Teknik untuk membuat kemasan vakum tersebut sudah digunakan untuk aplikasi MEMS, dan dapat diadopsi untuk membuat pembuatan komunikasi LOS antara antena CNT yang layak. Namun, pantulan dari permukaan substrat dan bahan pengemas mengganggu daya transmisi LOS. Dalam model saluran kami memperhitungkan refleksi multipath dari semua 6 permukaan kemasan serta kebisingan termal digabungkan dengan sinyal yang diterima dari transmisi LOS.

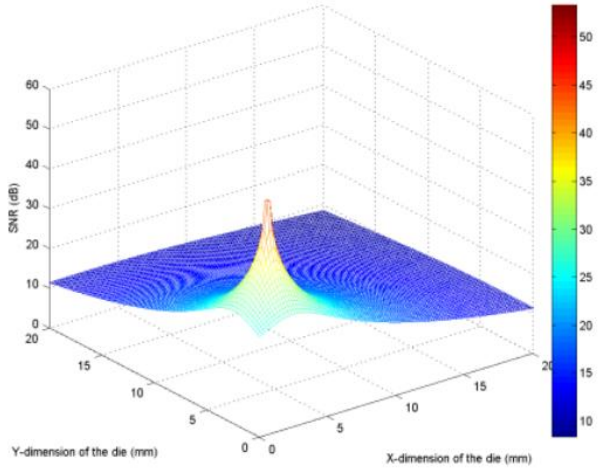


Gambar 7.11. Model saluran multi-jalur untuk tautan nirkabel on-chip

Gambar 7.11 menunjukkan transmisi multijalur sinyal dari pemancar ke penerima dengan semua kemungkinan sinar pantul dari empat dinding, langit-langit, dan tanah. Total daya yang diterima diberikan oleh

$$P_R = \frac{A_R}{4\Pi} \cdot P_T \cdot \left[\begin{array}{r} G_{T,LOS} \frac{e^{-j\frac{2\Pi}{\lambda}R_{LOS}}}{R_{LOS}} + \Gamma_1 G_{T1} \frac{e^{-j\frac{2\Pi}{\lambda}R_1}}{R_1} + \Gamma_2 G_{T2} \frac{e^{-j\frac{2\Pi}{\lambda}R_2 1}}{R_2} + \\ + \Gamma_3 G_{T3} \frac{e^{-j\frac{2\Pi}{\lambda}R_3}}{R_3} + \Gamma_4 G_{T1} \frac{e^{-j\frac{2\Pi}{\lambda}R_4}}{R_1} + \Gamma_{ceiling} G_{T,ceiling} \frac{e^{-j\frac{2\Pi}{\lambda}R_{ceiling}}}{R_{ceiling}} + \\ + \Gamma_{ground} G_{T,ground} \frac{e^{-j\frac{2\Pi}{\lambda}R_{ground}}}{R_{ground}} \end{array} \right]^2 , \quad (7.9)$$

di mana *GT,LOS* adalah gain antena pemancar sepanjang *LOS*, yang ditunjukkan -5dB. AR adalah area antena penerima dan *RLOS* adalah jarak LOS antara pemancar dan penerima. R_1, R_2, R_3, R_4, $R_{ceiling}$ dan R_{ground} adalah jarak sepanjang jalur pantulan yang berbeda. Γ_1, Γ_2, Γ_3, Γ_4, $\Gamma_{ceiling}$ dan Γ_{ground} merupakan koefisien refleksi pada permukaan substrat dan kemasan yang diperoleh. G_{T1}, G_{T2}, G_{T3}, G_{T4}, G_T, *ceiling* dan $G_{Tground}$ adalah penguatan antena sepanjang arah jalur pantulan yang semuanya kurang dari -10dB. Karena koefisien pantulan yang rendah dari bahan kemasan dan perolehan arah antena yang tinggi, hanya pantulan primer yang dipertimbangkan dalam model ini. Pemantulan berikutnya akan semakin mengurangi daya pemantulan dan dapat diabaikan dalam kasus ini. Daya derau termal diberikan oleh



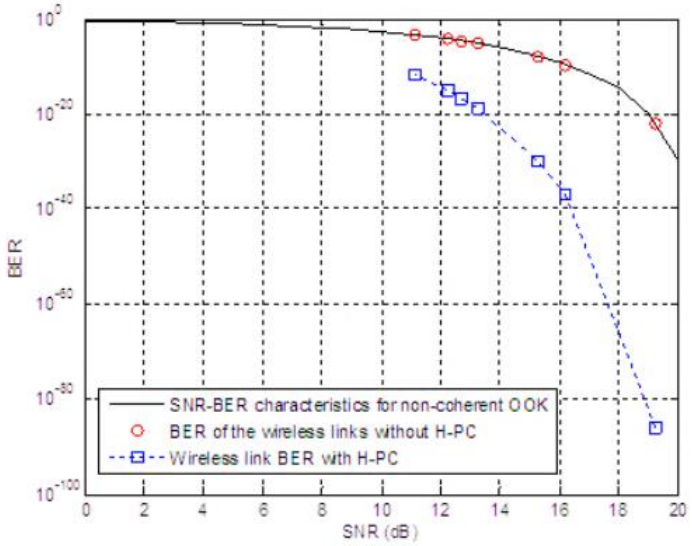
Gambar 7.12. SNR di atas area mati karena radiasi multipath dari pemancar ditempatkan di subnet pertama di pusatnya (X=2,5 mm, Y =2,5 mm)

$$N_0 = k \cdot T_0 \cdot F = k \cdot T_0 \cdot \left(\frac{T_{antenna}}{T_0} + F_r \right) \quad (7.10)$$

di mana k adalah konstanta Boltzmann, T0 adalah suhu ruangan yang diambil sebagai 290K, $T_{antenna}$ adalah suhu antena yang diasumsikan 330K, dan Fr adalah angka kebisingan penerima 4dB [38]. Kopling kebisingan peralihan chip ke saluran nirkabel dapat diabaikan karena saluran nirkabel berada dalam pita frekuensi sangat tinggi beberapa THz. Oleh karena itu, Rasio Signal-to-Noise (SNR) diberikan oleh

$$SNR = \frac{P_R}{N_0} \quad (7.11)$$

Gambar 7.12 menunjukkan variasi SNR pada area die 20mm 20mm untuk posisi tetap pemancar. Pemancar dalam hal ini ditempatkan di pusat subnet di sudut dekat pada gambar 12. SNR yang diterima maksimum di dekat pemancar dan secara bertahap berkurang dengan jarak. Meskipun ada pantulan multipath dari substrat dan dinding kemasan, daya pantul dapat diabaikan sebagaimana ditunjukkan oleh variasi kuadrat dari SNR yang diterima dengan jarak pada Gambar 7.12. Hal ini disebabkan oleh rendahnya koefisien pantulan dari permukaan pemantul dan direktivitas antena yang tinggi yang meredam radiasi dalam arah selain LOS. Karakteristik SNR vs bit-error-rate (BER) untuk tautan nirkabel sesuai dengan skema modulasi yang diadopsi, yang merupakan OOK non-koheren.



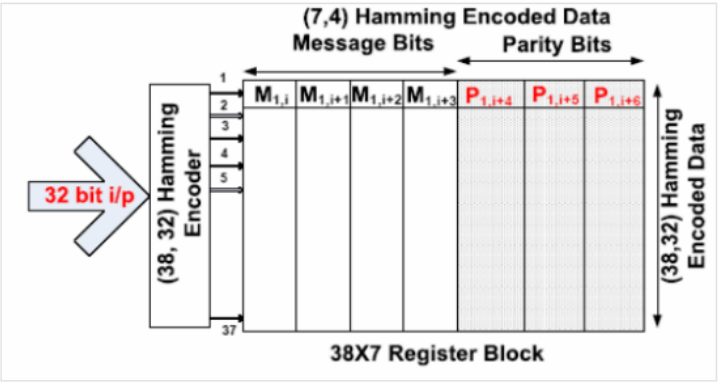
Gambar 7.13. Plot SNR vs BER dari saluran nirkabel dengan dan tanpa pengkodean

Gambar 7.13 menunjukkan variasi BER dengan SNR untuk penerima OOK yang tidak koheren. Konfigurasi khusus dari tautan nirkabel dibuat untuk kinerja jaringan yang optimal dengan anil yang disimulasikan menempatkan 24 tautan antara subnet tertentu di WiNoC. SNR dan karenanya BER yang sesuai dengan tautan tersebut juga ditandai dengan lingkaran di plot. SNR dan karenanya BER pada 24 tautan tersebut bervariasi karena setiap tautan memiliki panjang yang berbeda yang mengakibatkan hilangnya jalur yang berbeda dan pola radiasi yang dipantulkan. Namun, beberapa dari 24 tautan memiliki SNR yang sama dan karenanya muncul

sebagai titik yang sama di plot. Seperti dapat dilihat BER tertinggi untuk tautan nirkabel pada chip sekitar 4times10−4. Karena ini adalah BER tertinggi di antara semua link nirkabel, ini dapat disebut sebagai BER efektif dari WiNoC. BER efektif dari WiNoC ini jauh lebih tinggi daripada BER dari sambungan kabel, yang biasanya sekitar 10-15 atau kurang [6]. Oleh karena itu, kami mengusulkan penggunaan kode koreksi kesalahan multiple/burst yang kuat untuk saluran nirkabel. Pada sub-bagian berikutnya kami menjelaskan ECC berbasis kode produk untuk meningkatkan keandalan tautan nirkabel.

Kode Produk yang Diusulkan untuk Tautan Nirkabel

Untuk mencapai koreksi acak dan burst-error simultan pada tautan nirkabel, kode produk berbasis kode Hamming sederhana (H-PC). Penelitian sebelumnya menunjukkan bahwa kode produk yang dirancang dari kode Hamming koreksi kesalahan tunggal sederhana dalam 2 dimensi dapat bekerja lebih baik daripada beberapa kode koreksi kesalahan seperti kode Bose-Chaudhuri-Hocquenghem (BCH) atau kode Reed-Solomon (RS) di hal trade-off antara kinerja keseluruhan dan overhead. Di sini kami menunjukkan bahwa kode produk sederhana dapat digunakan untuk mencapai beberapa koreksi kesalahan serta koreksi kesalahan ledakan data yang dikirimkan melalui tautan nirkabel.



Gambar 7.14. Struktur skematis dari encoder H-PC yang diusulkan

Mengacu pada gambar 7.14 14 secara skematis menunjukkan struktur kode produk. Mari kita asumsikan ukuran flit k1 bit dan blok k2 bit tersebut. *(n1, k1)* Pengkodean Hamming dilakukan dalam dimensi spasial pada flit. Setiap *(n1, k1)* flit yang dikodekan Hamming kemudian dikodekan menggunakan *(n2, k2)* pengkodean Hamming dilakukan dalam dimensi waktu untuk memberikan kode produk *(n1 n2, k1 k2).* Dalam bab ini, kami memilih (38, 32) kode Hamming yang dipersingkat dalam dimensi spasial untuk menyandikan keseluruhan 32 bit pada satu waktu. Dalam dimensi waktu (7, 4) kode Hamming dipilih untuk meminimalkan latency dan buffering overhead menyimpan blok ukuran yang lebih besar. Setiap kode yang lebih besar akan menyebabkan kebutuhan buffering yang lebih tinggi pada node nirkabel di WiNoC. Dekoder kode produk menggunakan teknik decoding baris-kolom di mana pertama (38, 32) dekoder Hamming beroperasi pada kolom yang diterima dari blok dan kemudian (7, 4) dekoder Hamming mendekode baris untuk mengembalikan 4 yang diterima 32 bit terbang. Teknik decoding ini disebut sebagai decoding kolom-pertama. Untuk menutupi hukuman

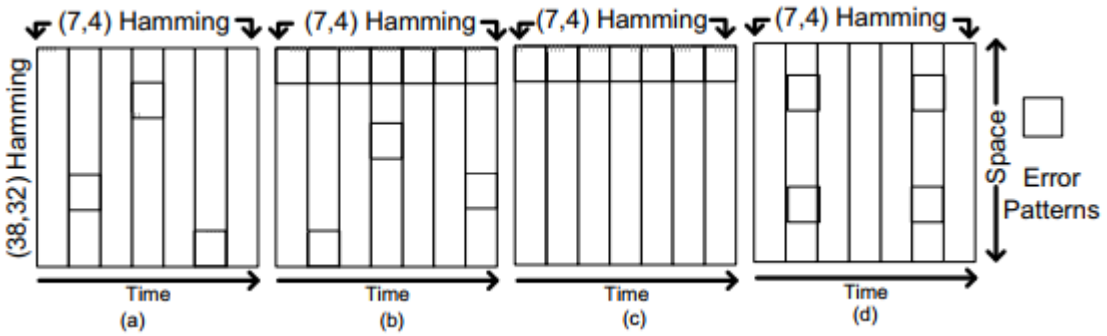
latensi decoding, decoder dirancang sedemikian rupa sehingga (38, 32) decoder Hamming beroperasi pada flit yang diterima saat mereka tiba dan 32 paralel (7, 4)

Dekoder Hamming kemudian beroperasi secara paralel pada bit 32 bit yang diterima. Ini meminimalkan overhead latensi dari dekoder H-PC.

Sisa BER dari Saluran Nirkabel dengan H-PC

Untuk memperkirakan efektivitas skema pengkodean yang diusulkan, kami melakukan analisis BER sisa setelah menerapkan ECC. Analisis ini didasarkan pada kejadian error bobot terkecil yang tidak dapat dikoreksi oleh kode produk, karena kejadian tersebut akan mendominasi BER pada SNR tinggi.

Gambar 7.15 menunjukkan berbagai pola kesalahan yang dapat diperbaiki dan tidak dapat diperbaiki dalam blok berukuran n1 n2 untuk teknik pendekodean baris-kolom. Skenario yang ditunjukkan pada Gambar 7.15(a) memiliki semburan spasial sepanjang flit tertentu yang diwakili oleh kolom yang diarsir, serta kesalahan acak bit tunggal pada flit lainnya. Hal ini dapat diperbaiki jika decoding kolom dilakukan terlebih dahulu diikuti oleh decoding baris, seperti pada decoding kolom-pertama yang diadopsi. Penguraian kode kolom akan memperbaiki semua kesalahan acak tetapi tidak ledakan, yang, bagaimanapun, akan diperbaiki dengan pengodean baris. Untuk skenario pada Gambar 7.15(b) dengan ledakan waktu tunggal dan beberapa kesalahan acak, skema decoding kolom pertama akan memperbaiki semua kesalahan jika pola kesalahan yang tidak dapat diperbaiki pada kolom menghasilkan pola kesalahan yang dapat diperbaiki pada baris setelah decoding kolom.



Gambar 7.15. (a)-(c): Berbagai pola kesalahan yang dapat diperbaiki. (d) Pola yang tidak dapat diperbaiki

Namun kasus ini tidak mungkin terjadi karena masing-masing antena bertanggung jawab untuk mentransmisikan beberapa bit dalam flit yang sama sebelum mentransmisikan bit dari flit yang lain. Kasus yang ditunjukkan pada Gambar 7.15(c) dengan semburan di setiap arah dapat dikoreksi sepenuhnya, karena decoding kolom akan mengoreksi semburan dalam waktu kecuali bit kiri atas. Pola yang dihasilkan hanyalah semburan dalam ruang, yang dapat dikoreksi dengan decoding baris. Pola kesalahan persegi panjang seperti yang ditunjukkan pada Gambar 7.15(d), tidak dapat dikoreksi dengan pendekodean baris-kolom karena kesalahan ganda terjadi pada baris dan kolom. Ini adalah peristiwa probabilitas tertinggi karena hanya 4 bit yang salah di seluruh blok yang menyebabkan pola kesalahan yang tidak dapat diperbaiki. Probabilitas kejadian ini diberikan oleh

$$P_{rectangle} = N_{rectangle} \cdot \epsilon^4 \left(1 - \epsilon^{n_1 \times n_2 - 4}\right), \quad (7.12)$$

di mana $N_{rectangle}$ adalah jumlah pola persegi panjang yang mungkin dalam sebuah blok berukuran $n1 \times n2$ dan β adalah BER tanpa pengkodean. $N_{rectangle}$ dihitung sebagai

$$N_{rectangle} = \sum_{l=2}^{n_1} \sum_{b=2}^{n_2} (n_1 - l + 1) \cdot (N - 2 - b + 1). \quad (7.13)$$

Pada SNR tinggi dimana peristiwa probabilitas tertinggi mendominasi BER, sisa BER dengan kode produk diberikan oleh

$$\epsilon_{PC} = \left(\frac{1}{n_1 n_2}\right) P_{rectangle}. \quad (7.14)$$

SNR baru vs BER sisa pada link nirkabel tertentu ditunjukkan pada Gambar 7.13 setelah menerapkan kode produk dengan kotak pada garis putus-putus. Efek dari kode produk secara signifikan menurunkan sisa BER. BER kasus terburuk pada link dengan SNR terendah adalah 1,99 10−12. Pengurangan BER ini memberikan tingkat kehandalan yang lebih tinggi dibandingkan dengan sistem uncoded. Selain itu, BER kasus terburuk dari saluran nirkabel menjadi sebanding dengan BER di tautan kabel dengan menggunakan skema H-PC.

Pengodean Kontrol Kesalahan untuk Tautan Wireline

Di subnet, transmisi data dilakukan melalui tautan kabel. Telah diketahui dengan baik bahwa dengan geometri yang menyusut, sambungan kabel akan semakin terpapar ke berbagai sumber kebisingan transien yang memengaruhi integritas sinyal dan keandalan sistem. Crosstalk yang bergantung pada data antara kabel yang berdekatan juga merupakan sumber utama kebisingan sementara tersebut. Karena ukuran fitur yang menyusut dalam teknologi masa depan, tingkat kesalahan transien diperkirakan akan meningkat beberapa kali lipat. Karena kesalahan ini belum tentu berkorelasi, tingkat kegagalan yang lebih tinggi dapat menyebabkan kesalahan banyak bit yang tidak berkorelasi dalam blok data.

Penghindaran crosstalk bersama dan beberapa kode koreksi kesalahan telah diusulkan. Kode-kode ini menghindari crosstalk kasus terburuk antara kabel yang berdekatan serta memperbaiki hingga tiga kesalahan acak dalam satu lompatan dan dapat mendeteksi empat kesalahan. Ide yang mendasari di balik skema pengkodean ini adalah bahwa jarak Hamming antara dua kata kode *Hsiao Single Error Correcting* dan *Double Error Detecting* (SEC-DED) adalah 4. Duplikasi bit yang dikodekan ini menghasilkan penggandaan jarak Hamming minimum antara kata kode menjadi 8. Dengan demikian dimungkinkan untuk memperbaiki 3 kesalahan dalam kata kode dan secara bersamaan mendeteksi kejadian 4 kesalahan. Menduplikasi bit dan menyisipkannya di samping garis bit asli juga berfungsi untuk menghindari crosstalk kasus terburuk dengan menghilangkan transisi berlawanan di kedua sisi kabel tertentu. Skema pengkodean ini disebut *sebagai Joint Crosstalk Avoidance Triple Error Correction* dan *Simultaneous Quadruple Error Detection Code* (JTEC-SQED).

Sisa BER dengan JTEC-SQED

Untuk menghitung probabilitas kesalahan kata residual untuk skema JTEC-SQED, mari kita asumsikan bahwa jumlah bit dalam *flit* adalah *2n + 2*, di mana ada 2 salinan dari *(n + 1, k)*

codeword SEC-DED. Karena JTEC-SQED dapat memperbaiki atau mendeteksi hingga empat kesalahan, batas bawah probabilitas decoding yang benar dapat diperoleh sebagai

$$P_{correct} \geq P(2n+2,0) + \ldots + P(2n+2,4), \tag{7.15}$$

di mana P(i, j) adalah probabilitas j kesalahan acak dalam i bit yang diberikan oleh

$$P(i,j) = \binom{i}{j} \epsilon^j (1-\epsilon)^{i-j} \tag{7.16}$$

Himpunan kata yang diterjemahkan dengan benar melengkapi himpunan kesalahan kata sisa. Oleh karena itu probabilitas kesalahan kata residual dapat dihitung menggunakan

$$P_{JTEC-SQED} = 1 - P_{orrect}, \tag{7.17}$$

di mana PJT EC−S QED adalah probabilitas kesalahan kata residual di hadapan pengkodean dan Porrect adalah probabilitas penguraian kode yang benar. Menggunakan Persamaan (7.9) dan (7.11) probabilitas kesalahan kata residual dari skema JTEC-SQED untuk nilai β yang kecil dapat didekati sebagai:

$$P_{JTEC-SQED} = \binom{2n+2}{5} \cdot \epsilon^5 \tag{7.18}$$

Penggabungan pengkodean kontrol kesalahan meningkatkan keandalan saluran komunikasi karena menjadi kuat terhadap malfungsi transien. Peningkatan keandalan dengan menggabungkan pengkodean dapat diterjemahkan ke dalam pengurangan ayunan tegangan pada kabel interkoneksi karena dapat mentolerir margin kebisingan yang lebih rendah. Hal ini menghasilkan penghematan dalam disipasi energi karena tergantung secara kuadratik pada ayunan tegangan. Pada bagian ini kami menghitung penguatan ini dengan memodelkan reduksi ayunan tegangan sebagai fungsi dari peningkatan kemampuan koreksi kesalahan. Efek kumulatif dari semua sumber derau UDSM transien dapat dimodelkan sebagai tegangan derau Gaussian tambahan VN dengan varians σ2. Dengan menggunakan model ini, BER β bergantung pada tegangan ayunan Vdd menurut

$$\epsilon = Q \cdot \left(\frac{V_{dd}}{2\sigma} \right) \tag{7.19}$$

di mana fungsi-Q diberikan oleh

$$Q(x) = \frac{1}{\sqrt{2\Pi} \int_x^{\infty} e^{\frac{y^2}{2}} dy}. \tag{7.20}$$

Probabilitas kesalahan kata adalah fungsi dari BER saluran ϵ. Jika *PUNC(ϵ)* adalah probabilitas sisa kesalahan kata dalam kasus yang tidak dikodekan dan *PECC (ϵ)* adalah probabilitas sisa kesalahan kata dengan pengkodean kontrol kesalahan, maka $PECC(\epsilon) \leq PUNC(\epsilon)$ diinginkan. Menggunakan Persamaan (7.13), kita dapat mengurangi tegangan suplai dengan adanya pengkodean ke *Vdd*, diberikan oleh

$$\widehat{V}_{dd} = V_{dd} \frac{Q^{-1}(\widehat{\epsilon})}{Q^{-1}(\epsilon)}. \tag{7.21}$$

Dalam Persamaan (7.21), Vdd adalah tegangan suplai nominal tanpa adanya pengkodean apa pun, V_{dd} adalah ayunan tegangan yang dikurangi dengan pengkodean dan ^$\hat{\epsilon}$ adalah BER sehingga

$$P_{ECC}(\widehat{\epsilon}) = P_{UNC}(\epsilon). \quad (7.22)$$

Penggunaan ayunan tegangan yang lebih rendah membuat probabilitas pola kesalahan multi-bit lebih tinggi, mengharuskan penggunaan beberapa kode koreksi kesalahan untuk mempertahankan probabilitas kesalahan kata yang sama dengan kasus yang tidak dikodekan. Karena skema JTEQ-SQED dapat mengurangi ayunan tegangan paling banyak dan memperbaiki jumlah kesalahan tertinggi dalam satu lompatan, skema ini dipilih untuk tautan wireline dari WiNoC.

Dengan menggunakan JTEC-SQED pada tautan wireline NoC, dimungkinkan untuk mempertahankan tingkat keandalan yang sama seperti tanpa ECC apa pun, tetapi dengan disipasi energi yang lebih rendah karena ayunan tegangan yang berkurang dan kapasitansi crosstalk pada kabel. Untuk link wireless seperti pada Gambar 7.13 BER tanpa ECC cukup tinggi. Namun, dengan H-PC dimungkinkan untuk mencapai BER yang sebanding dengan NoC wireline tanpa ECC. Kami menyelidiki kinerja WiNoC dengan skema pengkodean terpadu menggunakan JTEC-SQED untuk tautan wireline dan H-PC untuk tautan nirkabel untuk mencapai BER yang sama dengan NoC yang sepenuhnya wireline tanpa ECC. Pada bagian berikut kami menyajikan karakteristik kinerja dan disipasi energi WiNoC dengan skema pengkodean terpadu dan membandingkannya dengan NoC wireline.

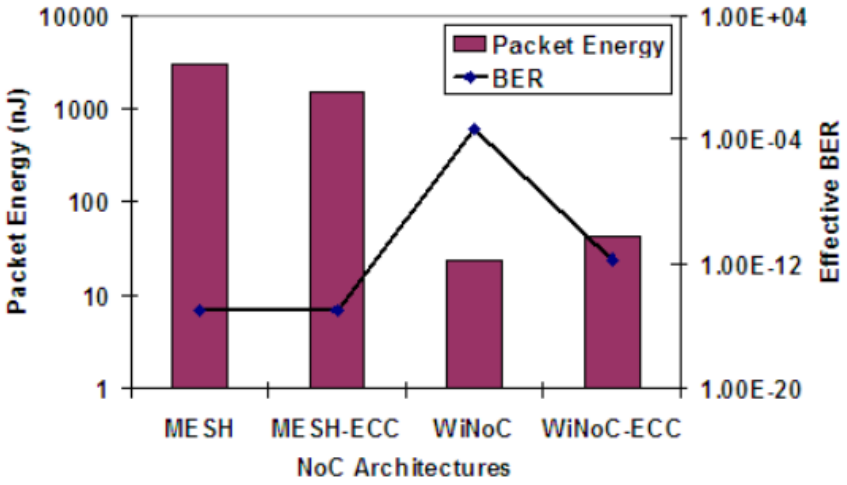
7.6 HASIL PERCOBAAN

Untuk mengkarakterisasi kinerja skema pengkodean yang diusulkan dalam WiNoC, kami mempertimbangkan sistem yang terdiri dari 256 inti. Mengikuti prinsip desain untuk throughput tertinggi diWiNoC dibagi menjadi 16 subnet masing-masing dengan 16 core, dan 24 link nirkabel didistribusikan di antara subnet. Ukuran level atas dan bawah dari hirarki WiNoC dipilih sama agar tidak membuat salah satunya lebih besar dari yang lain sehingga level tersebut menjadi bottleneck. Inti dalam setiap subnet terhubung dengan link wireline mengikuti topologi mesh. Kami mengasumsikan ukuran cetakan 20mm 20mm. Arsitektur switch dan hub.

Sakelar jaringan, hub, dan codec untuk ECC disintesis dari desain level RTL menggunakan pustaka sel standar 65 nm dari CMP (www.cmp.imag.fr), menggunakan Synopsys Design Vision dan dengan asumsi frekuensi clock 2,5 GHz. Disipasi energi pada kabel diperoleh dari CADENCE Spectre. Disipasi energi pada link nirkabel diperoleh dari Persamaan (7.3). WiNoC disimulasikan menggunakan simulator siklus yang akurat yang memodelkan progres data flits secara akurat per siklus clock yang memperhitungkan flits yang mencapai tujuan dan juga yang dijatuhkan.

Gambar 7.16 menunjukkan disipasi energi paket dari WiNoC dengan dan tanpa skema ECC terpadu yang diusulkan. Disipasi energi paket adalah energi yang hilang dalam mentransfer satu paket dari sumber ke tujuan. Sebagai perbandingan, disipasi energi paket dalam NoC wireline mesh lengkap dengan dan tanpa ECC juga ditampilkan. ECC yang

digunakan dalam link wireline subnet adalah JTEC-SQED karena merupakan skema pengkodean yang paling hemat energi. BER yang efektif pada tautan komunikasi dalam kasus yang sesuai juga ditampilkan. Untuk mesh dengan JTEC-SQED, BER-nya sama seperti di mesh tanpa ECC karena ayunan tegangan berkurang pada link sesuai dengan Persamaan (7.15). Untuk WiNoCs, BER pada saluran nirkabel jauh lebih tinggi daripada tautan kabel dan karenanya ini menunjukkan BER yang efektif pada tautan komunikasi. Energi paket wireline mesh berkurang karena JTEC-SQED, karena ayunan tegangan pada link wireline dapat dikurangi secara signifikan menurut Persamaan (7.15). Selain itu, kapasitansi kopling crosstalk pada kabel berkurang. Penghematan energi karena kedua faktor ini lebih banyak daripada biaya overhead karena codec dan tautan yang berlebihan. Keandalan keseluruhan pada kabel sama dalam kedua kasus karena pengurangan disipasi energi diproyeksikan untuk BER yang sama pada kabel. Untuk WiNoC tanpa ECC kasus terburuk BER jauh lebih tinggi karena SNR rendah. Tetapi dengan pengkodean HPC yang kuat pada saluran nirkabel, BER saluran nirkabel dikurangi menjadi sekitar 10-12.



Gambar 7.16. Disipasi energi paket dan BER kasus terburuk saluran untuk WiNoC dan arsitektur mesh dengan dan tanpa ECC

Oleh karena itu, keandalan WiNoC secara keseluruhan ditingkatkan dengan skema pengkodean. Overhead dari skema H-PC dan JTEC-SQED dipertimbangkan dan kami menemukan bahwa ada peningkatan energi paket karena overhead codec dari ECC dibandingkan dengan WiNoC tanpa pengkodean apa pun. Namun, dengan sedikit peningkatan energi paket ini, kami dapat mencapai BER keseluruhan yang sebanding pada WiNoC seperti pada NoC wireline mesh sepenuhnya. Disipasi energi paket dari WiNoC dengan ECC bagaimanapun masih tetap beberapa kali lipat lebih rendah dibandingkan rekanan wireline lengkap karena sambungan nirkabel jarak jauh berdaya rendah.

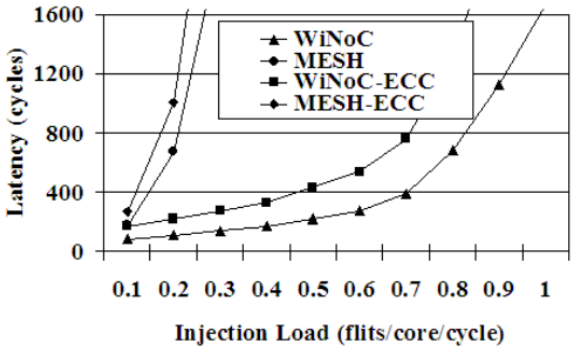
Kami juga memperkirakan karakteristik waktu WiNoC dibandingkan dengan wireline mesh. Latensi pesan end-to-end dari wireline mesh dengan JTEC-SQED lebih tinggi daripada mesh tanpa ECC. Ini karena codec ECC menambahkan overhead ke jalur kritis di sakelar NoC. WiNoC tanpa ECC mencapai latensi yang jauh lebih rendah dibandingkan dengan jaringan

kabel karena keuntungan dari pintasan nirkabel jarak jauh yang dimasukkan ke dalam jaringan serta pembagian hierarki NoC. Karena pintasan nirkabel di WiNoC, jumlah hop rata-rata antar core jauh lebih sedikit dibandingkan dengan mesh dengan ukuran yang sama. Oleh karena itu, seperti yang ditunjukkan pada [8] kinerja WiNoC jauh lebih baik dibandingkan dengan NoC wireline mesh. Namun, mirip dengan skema JTEC-SQED, codec H-PC juga menambahkan overhead pengaturan waktu ke tautan nirkabel. Rancangan encoder seperti yang diuraikan sebelumnya membutuhkan pengkodean (38, 32) pada setiap 32bit flit yang kemudian disimpan hingga 4 flits diterima. Semua 4 flit kemudian dikodekan dengan 38 encoder Hamming paralel (7, 4). Oleh karena itu, overhead pembuat enkode H-PC sama dengan penundaan satu (38, 32) pembuat enkode Hamming dan satu (7, 4) pembuat enkode Hamming karena semua pembuat enkode (7, 4) beroperasi secara paralel. Penundaan dari berbagai tahapan skema H-PC dan JTEC-SQED ditunjukkan pada Tabel 7.4.

Penalti latensi karena tingkat kode (32×4)/(38×7) = 0,48 juga diperhitungkan. Namun dalam tautan kabel, siklus tambahan tidak diperlukan karena laju kode karena bit yang berlebihan dapat ditransfer dalam siklus yang sama dengan kabel tambahan.

Tabel 7.4. Delay untuk Setiap Skema Pengodean

Coding Sceme	Codec Delay (ps)	
	Encoder	Decoder
H-PC	150	354
JTEC-SQED	133	315



Gambar 7.17. Karakteristik latensi arsitektur mesh dan WiNoC dengan dan tanpa ECC

Gambar 7.17 menunjukkan keseluruhan karakteristik latency dari WiNoC dan wireline mesh dengan dan tanpa coding. Karena tingkat kode yang rendah dari H-PC serta codec overhead, latensi keseluruhan WiNoC dengan pengkodean lebih tinggi daripada WiNoC tanpa pengkodean. Namun, bahkan dengan pengkodean, latensi WiNoC jauh lebih sedikit dibandingkan dengan NoC wireline mesh tanpa pengkodean apa pun.

Codec memperkenalkan komponen perangkat keras tambahan dan karenanya juga memerlukan overhead area silikon. Tabel 7.5 merangkum overhead area dari masing-masing skema pengkodean. Area yang dilaporkan adalah area codec yang diperlukan per port switch atau hub.

Tabel 7.5. Area Overhead dari Codec untuk Setiap Skema Pengkodean

Coding Scheme	Area ($\mu m^{2)}$
H-PC	29710
JTEC-SQED	11055

7.7 KESIMPULAN

Wireless *Network-on-Chip* telah muncul sebagai salah satu dari banyak teknologi interkoneksi alternatif untuk chip multi-core masa depan. Namun, keandalan tautan nirkabel on-chip jauh lebih sedikit dibandingkan dengan interkoneksi kabel. Di sini kami menyajikan arsitektur WiNoC yang terinspirasi sifat hibrida hierarkis bersama dengan mekanisme ECC terpadu yang dengannya kami dapat memulihkan keandalan tautan nirkabel ke interkoneksi kabel dan juga mengurangi pemborosan energi pada interkoneksi kabel. Hal ini ditunjukkan bahwa dengan link nirkabel yang ditingkatkan ECC pada chip memungkinkan untuk mencapai disipasi energi yang lebih rendah, latensi yang lebih rendah dan komunikasi data on-chip yang hampir sama andalnya dibandingkan dengan NoC wireline multi-hop tradisional. Dengan demikian mengarah ke paradigma komputasi multi-core yang berkelanjutan dan andal.

BAB 8
ALGORITMA EVOLUSIONER
UNTUK MAKSIMALKAN JARINGAN NIRKABEL

Teknologi jaringan sensor nirkabel (WSN) menjanjikan potensi tinggi untuk mengatasi tantangan lingkungan dan memantau serta mengurangi energi dan emisi gas rumah kaca. Memang, WSN telah berhasil digunakan dalam aplikasi seperti bangunan cerdas, jaringan cerdas dan sistem kontrol energi, transportasi dan logistik, dan pertanian presisi. Semua aplikasi ini umumnya membutuhkan pertukaran data dalam jumlah besar dan lokalisasi node sensor. Kedua tugas ini bisa sangat haus energi. Karena node sensor biasanya ditenagai oleh baterai kecil, strategi penghematan energi yang tepat harus digunakan untuk memperpanjang masa pakai WSN dan membuat penggunaannya menarik dan efektif. Untuk tujuan ini, studi tentang algoritma kompresi data yang cocok untuk penyimpanan yang dikurangi dan sumber daya komputasi dari sebuah node sensor, dan eksplorasi teknik lokalisasi node bertujuan untuk memperkirakan posisi semua node sensor dari sebuah WSN dari pengetahuan tentang lokasi yang tepat dari node sensor. sejumlah terbatas dari node ini, telah menarik minat yang besar dalam beberapa tahun terakhir. Dalam bab ini, kita membahas bagaimana algoritma evolusioner multi-objektif dapat berhasil dieksploitasi untuk menghasilkan kompresor data yang sadar energi dan untuk memecahkan masalah lokalisasi node. Hasil simulasi menunjukkan bahwa, dalam kedua tugas tersebut, solusi yang dihasilkan oleh proses evolusi mengungguli pendekatan paling menarik yang baru-baru ini diajukan dalam literatur.

8.1 PENDAHULUAN

Wireless Sensor Network (WSN) adalah kumpulan node yang diatur ke dalam jaringan kooperatif. Setiap node adalah perangkat kecil yang terdiri dari tiga unit dasar: unit pemrosesan dengan memori terbatas dan daya komputasi, unit penginderaan untuk akuisisi data dari lingkungan sekitar dan unit komunikasi, biasanya transceiver radio, untuk mengirimkan data ke pusat titik pengumpulan, dilambangkan dengan sink node atau base station. Biasanya, node ditenagai oleh baterai kecil yang umumnya tidak dapat diubah atau diisi ulang.

WSN saat ini mewakili teknologi dewasa yang berdampak pada beberapa aplikasi dunia nyata. Sebagai contoh, mari kita pertimbangkan bagaimana WSN mengubah pertanian klasik. Ketersediaan node sensor bertenaga baterai yang murah, yang dapat digunakan di area yang luas, memungkinkan petani, misalnya, untuk menggunakan alat bantu panduan elektronik untuk mengarahkan pergerakan peralatan secara lebih akurat dan untuk menyediakan pemosisian yang tepat untuk semua aksi peralatan dan aplikasi bahan kimia. Selanjutnya, sejumlah besar sensor yang berbeda memungkinkan pengumpulan ukuran parameter agronomi dan iklim secara real-time, sehingga memungkinkan untuk mendeteksi kondisi yang tidak diinginkan dan segera mengatasi efeknya. Secara umum, setelah disebarkan secara acak di area yang akan dipantau, node sensor dapat secara mandiri membangun topologi jaringan

ad-hoc. Konfigurasi ad-hoc merupakan fitur kunci dalam WSN karena memungkinkan penyebaran yang mudah dan cepat dan memberikan tingkat tertentu dari rekonfigurasi, kehandalan dan toleransi kesalahan.

Dalam aplikasi lingkungan tipikal dari WSN, setiap node sensor memantau parameter lingkungan dan menghasilkan aliran pengukuran yang harus ditransmisikan dari node sensor itu sendiri ke stasiun pangkalan, melalui komunikasi perutean multi-hop. Oleh karena itu, node yang bertindak sebagai router harus mengelola ukuran yang dikumpulkan oleh sensor di atas node itu sendiri, dan untuk menyimpan dan meneruskan ke sink baik ukuran ini maupun data yang berasal dari node lain. Karena komunikasi radio biasanya dianggap sebagai penyebab utama konsumsi daya, masa pakai node ini dan akibatnya keseluruhan WSN cenderung sangat singkat jika strategi yang tepat ditujukan untuk membatasi transmisi/penerimaan data sebanyak mungkin tidak dilakukan.

Kompresi data muncul sebagai alat yang sangat menarik dan efektif untuk mencapai tujuan ini. Algoritma kompresi data dapat secara luas diklasifikasikan menjadi dua kelas: algoritma lossless dan lossy. Algoritme lossless menjamin integritas data selama proses kompresi/dekompresi. Sebaliknya, algoritma lossy dapat menghasilkan hilangnya informasi, tetapi umumnya memastikan rasio kompresi yang lebih tinggi. Sayangnya, desain teknik kecil dari node sensor yang tersedia secara komersial mencegah penerapan beberapa skema kompresi data klasik yang membutuhkan sejumlah memori dan daya komputasi yang tidak tersedia di node ini. Selanjutnya, karena sensor pada node board biasanya cukup murah, ukuran parameter lingkungan yang dikumpulkan oleh sensor ini sering dipengaruhi oleh noise. Kebisingan ini meningkatkan entropi informasi dan karenanya menghambat algoritma kompresi lossless untuk mencapai rasio kompresi yang cukup besar.

Jadi, hanya untuk mentransmisikan noise, penggunaan algoritme kompresi lossless tidak memungkinkan peningkatan masa pakai node sensor terlalu banyak. Solusi yang ideal adalah dengan mengadopsi algoritma kompresi lossy pada node sensor di mana informasi yang hilang hanyalah noise. Dalam hal ini, kami dapat mencapai rasio kompresi yang tinggi tanpa kehilangan informasi yang relevan. Untuk tujuan ini, kami mengeksploitasi pengamatan bahwa data yang biasanya dikumpulkan oleh WSN sangat berkorelasi. Dengan demikian, perbedaan antara sampel berurutan umumnya cukup kecil. Jika hal ini tidak terjadi, kemungkinan besar sampel dipengaruhi oleh kebisingan. Untuk de-noise dan sekaligus kompres sampel, kami mengkuantisasi perbedaan antara sampel berturut-turut. Selanjutnya, untuk mengurangi jumlah bit yang diperlukan untuk mengkodekan perbedaan ini, kami mengadopsi skema Modulasi Kode Pulsa Diferensial (DPCM). Tentu saja, kombinasi yang berbeda dari parameter proses kuantisasi menentukan trade-off yang berbeda antara kinerja kompresi dan kehilangan informasi. Untuk menghasilkan satu set kombinasi optimal dari parameter ini, kami menggunakan pendekatan optimasi multi-objektif.

Optimasi multi-objektif adalah proses mengoptimalkan dua atau lebih tujuan secara bersamaan dengan kendala tertentu. Untuk masalah multi-tujuan nontrivial, solusi unik yang secara bersamaan mengoptimalkan setiap tujuan tidak dapat ditentukan. Memang, ketika mencari solusi, seseorang sampai pada tahap di mana optimalisasi lebih lanjut dari suatu tujuan menyiratkan bahwa tujuan lain menderita sebagai akibatnya. Sebuah solusi tentatif

disebut non-dominasi atau optimal Pareto jika tidak dapat digantikan oleh solusi lain yang meningkatkan tujuan tanpa memperburuk yang lain. Secara lebih formal, solusi x terkait dengan vektor kinerja u mendominasi solusi y terkait dengan vektor kinerja v jika dan hanya jika, i 1, ..., I , dengan I jumlah tujuan, ui berkinerja lebih baik dari, atau sama ke, vi dan i 1, ..., I sehingga ui bekerja lebih baik daripada vi, di mana ui dan vi masing-masing adalah elemen ke-i dari vektor u dan v. Himpunan solusi optimal Pareto dilambangkan sebagai depan Pareto. Dengan demikian, tujuan dari setiap algoritma multi-tujuan adalah untuk menemukan keluarga solusi yang merupakan pendekatan yang baik dari front Pareto. Algoritma Evolusi Multi-Tujuan (MOEA), yang digunakan dalam pekerjaan ini, mencapai tujuan ini dengan mensimulasikan proses evolusi alami. Tujuan pertama dari bab ini adalah untuk menyajikan bahwa penggunaan MOEA adalah pendekatan yang efektif untuk pembuatan kompresor untuk WSN dengan membandingkan solusi representatif dari pendekatan Pareto front dengan salah satu kompresor paling efektif untuk WSN yang baru-baru ini diusulkan di literature.

Setelah informasi terkompresi mencapai sink, itu diuraikan untuk aplikasi tertentu yang dipertimbangkan. Seringkali, terutama dalam domain lingkungan, proses elaborasi membutuhkan untuk mengetahui lokasi node di mana pengukuran tunggal telah dikumpulkan. Meskipun kesadaran lokasi dapat diaktifkan pada prinsipnya dengan menggunakan *Global Positioning System* (GPS) di papan node sensor, solusi ini tidak selalu layak dalam praktiknya, karena biaya dan konsumsi daya penerima GPS tidak dapat diabaikan. Selain itu, GPS tidak cocok untuk pemasangan di dalam dan di bawah tanah, dan adanya penghalang seperti dedaunan lebat atau gedung tinggi dapat mengganggu komunikasi luar ruangan dengan satelit.

Dalam skema ini, hanya beberapa node jaringan (node referensi atau jangkar) yang diberi posisi tepat melalui GPS atau penempatan manual, sementara semua node dapat memperkirakan jarak mereka ke node terdekat dengan menggunakan teknik pengukuran yang sesuai. Seperti Kekuatan Sinyal yang Diterima (RSS), Waktu Kedatangan (ToA), Perbedaan Waktu Kedatangan (TDoA). Dengan demikian, dengan asumsi bahwa koordinat simpul jangkar diketahui, dan mengeksploitasi pengukuran jarak berpasangan di antara simpul, skema lokalisasi berbutir halus bertujuan untuk menentukan posisi semua simpul non-jangkar dengan meminimalkan fungsi biaya (CF) dihitung sebagai kesalahan kuadrat antara perkiraan dan jarak antar node terukur yang sesuai. Tugas ini adalah masalah optimisasi nonconvex multivariabel yang telah terbukti menjadi NP-hard dan karena itu agak sulit untuk diselesaikan dengan menggunakan teknik tradisional. Terlebih lagi, pengukuran jarak berpasangan selalu dirusak oleh noise. Akhirnya, bahkan jika pengukuran jarak akurat, kondisi yang cukup untuk topologi agar dapat dilokalkan secara unik tidak mudah untuk diidentifikasi. Kami ingat bahwa jaringan dikatakan dapat dilokalkan jika hanya ada satu kemungkinan geometri yang kompatibel dengan data yang tersedia.

Dalam beberapa tahun terakhir, beberapa upaya telah dilakukan untuk mengurangi kompleksitas komputasi dari masalah optimasi nonconvex dengan merelaksasinya menjadi masalah optimasi cembung. Berbicara secara longgar, formulasi ulang relaksasi cembung bertujuan untuk menukar waktu komputasi untuk beberapa akurasi dalam perkiraan akhir. Dengan demikian, hasil metode santai sangat cocok untuk memecahkan masalah lokalisasi

dalam skala besar dan jaringan seluler. Di sisi lain, ada aplikasi praktis dari WSN yang tidak memerlukan solusi real-time yang sangat skalabel. Pertimbangkan, misalnya, aplikasi pertanian presisi yang kami sebutkan di atas: jelas lebih baik menghabiskan beberapa waktu tambahan untuk mendapatkan perkiraan koordinat simpul yang akurat daripada, misalnya, memberikan pupuk atau pestisida ke zona yang salah. dari lapangan yang dipantau. Selain itu, testbed jaringan sensor yang ada saat ini jarang disusun oleh lebih dari seratus node dan mobilitasnya terutama dimungkinkan untuk aplikasi keamanan dan pengawasan. Dengan demikian, kami tertarik pada skema lokalisasi yang menghasilkan estimasi akurat, meskipun kami sangat sadar bahwa skema tersebut mungkin tidak cocok untuk aplikasi yang menuntut skalabilitas tinggi dan membutuhkan operasi real-time.

Tujuan kedua dari bab ini adalah untuk menunjukkan bagaimana masalah optimisasi nonconvex dapat diatasi dengan mengadopsi MOEA yang memperhitungkan secara bersamaan selama proses evolusi baik CF dan kendala topologi tertentu yang disebabkan oleh pertimbangan konektivitas. Kendala ini sangat berguna untuk mengurangi masalah lokalisasi. Memang, jika jaringan tidak dapat dilokalkan, maka beberapa minimum CF akan muncul, dengan hanya satu di antaranya yang sesuai dengan geometri penyebaran yang sebenarnya. Dengan demikian, dalam pengaturan yang hampir tidak dapat dilokalkan, setiap algoritma lokalisasi akan menjadi sangat sensitif terhadap minima palsu ini, menghasilkan kesalahan lokalisasi yang sangat besar. Kami menyajikan bahwa penggunaan MOEA adalah pendekatan yang efektif untuk memecahkan masalah lokalisasi dengan membandingkan solusi representatif dari front Pareto yang didekati dengan dua pendekatan yang paling berkinerja yang baru-baru ini diusulkan dalam literatur.

Sebagai kesimpulan, tujuan utama dari bab ini adalah untuk membahas bagaimana MOEA dapat berhasil dieksploitasi untuk mengatasi dua masalah yang sangat relevan di WSN: untuk menghasilkan kompresor data lossy yang sadar energi dan untuk memecahkan masalah lokalisasi node. Bab ini disusun sebagai berikut. Bagian 8.2 membahas pekerjaan terkait, sedangkan Bagian 8.3 dan 8.4 menunjukkan usulan pendekatan berbasis MOEA untuk menghasilkan kompresor data yang efektif dan untuk memecahkan masalah lokalisasi, masing-masing. Di Bagian 8.5, kami menjelaskan MOEA yang digunakan dalam eksperimen kami. Bagian 8.6 dan 8.7 menunjukkan beberapa percobaan yang dilakukan dengan menggunakan pendekatan yang diusulkan. Secara khusus, kami menyoroti bagaimana solusi yang diperoleh dengan penerapan MOEA mengungguli beberapa pendekatan canggih yang baru-baru ini diusulkan dalam literatur. Akhirnya, Bagian 8.8 menarik beberapa kesimpulan.

8.2 PEKERJAAN TERKAIT

Pada bagian ini kami meninjau secara singkat pekerjaan yang terkait dengan kompresi data dan lokalisasi node di WSN.

Kompresi Data di WSN

Karena keterbatasan sumber daya yang tersedia di node sensor, kompresi data di WSN memerlukan algoritme yang dirancang khusus. Dua pendekatan telah diadopsi dalam literatur: untuk mendistribusikan biaya komputasi pada keseluruhan jaringan dan untuk mengaktifkan kompresi yang bekerja pada node tunggal secara independen dari yang lain.

Pendekatan pertama alami dalam WSN yang kooperatif dan padat, di mana node dapat berkolaborasi satu sama lain untuk melaksanakan tugas yang tidak dapat mereka lakukan sendiri. Selain itu, berkat topologi khusus dari WSN ini, data yang diukur oleh node tetangga berkorelasi: dalam skenario ini mungkin masuk akal bagi node sensor untuk bersama-sama memperkirakan fenomena yang sebagian besar berkorelasi, hanya dengan bertukar pesan. Skema *Distributed Source Coding* (DSC) dapat dianggap pengecualian sehubungan dengan skema yang diusulkan untuk pendekatan pertama. Memang, tidak seperti metode sebelumnya tidak memerlukan pertukaran pesan antar node. Di DSC, node sensor hanya mengirim output terkompresi mereka ke titik pusat yang bertanggung jawab untuk bersama-sama mendekode aliran yang disandikan. Sayangnya, asumsi terkuat di semua skema terdistribusi adalah bahwa node sensor dikerahkan secara padat sehingga pembacaannya sangat berkorelasi dengan tetangganya.

Berbeda dengan pendekatan terdistribusi, pendekatan kedua untuk kompresi data tidak mengeksploitasi korelasi pengukuran yang dikumpulkan dari node tetangga: setiap node hanya mengeksploitasi informasi lokalnya untuk mengompresi data. Hal ini dapat membuat pencapaian rasio kompresi yang memuaskan menjadi lebih sulit. Selanjutnya, itu bisa membutuhkan eksekusi instruksi dalam jumlah yang lebih besar. Nyatanya, penghematan daya hanya dapat dicapai jika eksekusi algoritme kompresi tidak memerlukan jumlah energi yang lebih besar daripada energi yang dihemat dalam mengurangi komunikasi. Memang, setelah menganalisis beberapa keluarga algoritme kompresi klasik, Barr dan Asanovic' menyimpulkan bahwa kompresi sebelum transmisi pada perangkat bertenaga baterai nirkabel sebenarnya dapat menyebabkan peningkatan konsumsi daya secara keseluruhan. Di sisi lain, algoritme kompresi standar ditujukan untuk menghemat penyimpanan dan bukan energi. Dengan demikian, strategi yang tepat harus diadopsi.

Contoh teknik kompresi yang diterapkan pada node tunggal mengadaptasi beberapa algoritme kompresi berbasis kamus yang ada dengan kendala yang dikenakan oleh sumber daya terbatas yang tersedia pada node sensor. Sebagai contoh, algoritma kompresi lossless adalah versi LZ77 yang sengaja diadaptasi, kode Eksponensial–Golomb dan LZW. Algoritma Lightweight Temporal Compression (LTC) adalah teknik kompresi lossy yang efisien dan sederhana untuk konteks pemantauan habitat. LTC memperkenalkan sejumlah kecil kesalahan ke dalam setiap pembacaan yang dibatasi oleh kenop kontrol: semakin besar batas kesalahan ini, semakin besar penghematan kompresi. Pada dasarnya LTC mirip dengan Run Length Encoding (RLE) dalam arti bahwa LTC mencoba merepresentasikan urutan panjang dari data yang mirip dengan satu simbol. Perbedaannya dengan RLE adalah bahwa sementara RLE mencari string dari simbol berulang, LTC mencari tren linier. Kami akan menggunakan LTC sebagai pembanding karena, sepengetahuan kami, LTC adalah algoritma kompresi lossy yang unik yang dirancang khusus untuk kompresi data lossy pada node sensor tunggal dari WSN. Selanjutnya, MOEA belum pernah diterapkan sebelumnya untuk menghasilkan kompresor lossy untuk WSN.

Lokalisasi Node di WSN

Masalah lokalisasi yang halus terbukti menjadi masalah NP-hard. Dalam literatur tiga pendekatan yang berbeda dapat ditemukan untuk mengatasi masalah tersebut, yaitu,

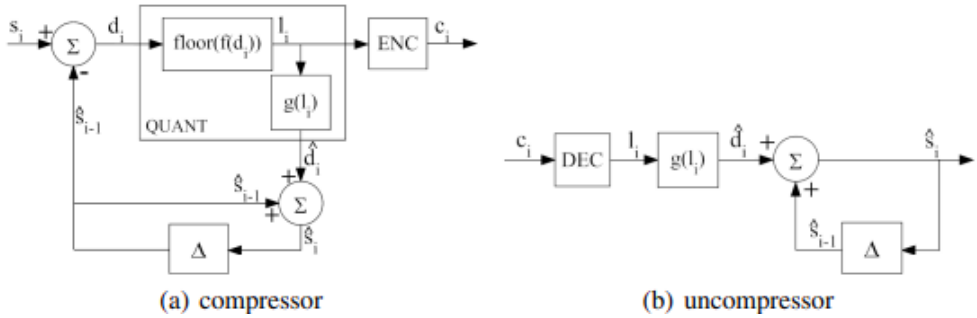
optimasi stokastik, penskalaan multidimensi, dan relaksasi cembung. Pendekatan pertama mencoba untuk menghindari minima lokal dengan beralih ke metode optimasi global, seperti mis. simulasi anil. Penskalaan multidimensi adalah teknik berbasis konektivitas yang, selain pengukuran jarak, mengeksploitasi pengetahuan tentang topologi jaringan; informasi ini memberikan kendala tambahan pada masalah, karena node dalam jangkauan komunikasi satu sama lain tidak dapat berjauhan secara sewenang-wenang. Pendekatan ketiga melonggarkan formulasi nonconvex asli untuk mendapatkan masalah *Semi-Definite Programming* (SDP) atau *Second-Order Cone Programming* (SOCP). Solusi global untuk santai, masalah cembung dapat kemudian diperoleh dengan upaya komputasi moderat dan merupakan solusi perkiraan untuk masalah nonconvex asli.

8.3 KOMPRESI DATA DI WSN: SOLUSI BERBASIS MOEA

Pada bagian ini kami menjelaskan pendekatan berbasis MOEA kami untuk menghasilkan kompresor data lossy yang menyadari energi di WSN. Secara khusus, Bagian ini memperkenalkan pernyataan masalah dengan mengingat beberapa prinsip kuantisasi dasar. Dalam Bagian juga kami menunjukkan ikhtisar dari pendekatan kami. Terakhir, Bagian terakhir menjelaskan representasi kromosom dan operator perkawinan.

Rumusan Masalah

Data lingkungan biasanya dicirikan oleh korelasi yang tinggi antara sampel yang bertetangga. Untuk jenis sinyal ini, metode kompresi diferensial terbukti sangat efektif. Jadi, untuk mengompresi data di WSN, kami mengadopsi versi skema DPCM yang diadaptasi secara sengaja, yang sering digunakan untuk kompresi sinyal audio digital. DPCM adalah anggota dari keluarga metode kompresi diferensial.



Gambar 8.1. Blok diagram (a) kompresor dan (b) uncompressor

Seperti yang ditujukan pada gambar. 8.1(a) dan 8.1(b) masing-masing menunjukkan diagram blok kompresor dan uncompressor kita. Mengenai kompresor, perbedaan umum di dihitung dengan mengurangkan nilai rekonstruksi terbaru sˆi−1 dari sampel saat ini si. Untuk menggunakan sˆi−1 daripada nilai aslinya si−1 menghindari masalah akumulasi kesalahan yang terkenal [34]. Oleh karena itu, perbedaan di adalah input ke blok kuantisasi QUANT. Misalkan S = S 0,..., S L−1 adalah himpunan sel S l, dengan l ∈ [0, L-1], yang membentuk partisi disjoint dan exhaustive dari domain input D (domain difference dalam kasus kita). Misalkan C = {y0,..., yL−1} adalah himpunan level yl ∈ S l. Blok floor(f (•)) mengembalikan indeks li dari sel S li yang menjadi miliknya. Indeks li adalah input ke blok g ( ), yang menghitung perbedaan

terkuantisasi dˆi, dan ke encoder entropi ENC, yang menghasilkan codeword biner ci. Karena sinyal lingkungan cukup halus dan oleh karena itu perbedaan kecil lebih mungkin terjadi daripada yang besar, hasil enkoder entropi akan bekerja secara khusus. Memang, pembuat enkode ini mengkode indeks yang lebih memungkinkan dengan jumlah bit yang lebih rendah. Pada uncompressor, codeword ci dianalisa oleh decoding block DEC yang mengeluarkan indeks li.

Indeks ini dijabarkan dengan blok g ( ) untuk menghasilkan dˆi, yang ditambahkan ke sˆi−1 untuk menghasilkan output sˆi.

Proses kuantisasi bergantung pada jumlah dan ukuran sel S l dan pada posisi level yl ∈ S l. Sejumlah kecil sel lebar memungkinkan pencapaian reduksi data yang tinggi, tetapi juga menghasilkan kesalahan rekonstruksi yang tinggi pada dekoder. Di sisi lain, jika jumlah selnya tinggi, kesalahan rekonstruksinya berkurang tetapi rasio kompresinya juga menurun. Dengan demikian, trade-off yang tepat antara kesalahan kompresi dan rekonstruksi harus ditemukan.

Gambaran Umum Pendekatan Kami

Dengan tujuan untuk menentukan satu set parameter kuantisasi yang optimal, kami mengadopsi MOEA yang secara bersamaan mengoptimalkan dua tujuan, yaitu, entropi H dan kesalahan kuadrat rata-rata MS E antara sampel terkuantisasi dan de-noise ideal (kami akan menunjukkan seperti itu ukur sebagai MS E∗).

Informasi entropi H memberikan ukuran tidak langsung dari kemungkinan rasio kompresi yang dapat diperoleh dan didefinisikan sebagai:

$$H = -\sum_{l=1}^{L} p_l \cdot \log_2(p_l) \tag{8.1}$$

di mana pl adalah fungsi massa probabilitas dari indeks kuantisasi l.

MS E menghitung distorsi antara dua sinyal dan didefinisikan sebagai

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(s_i^{(1)} - s_i^{(2)}\right)^2 \tag{8.2}$$

di mana N adalah jumlah sampel, dan $s_i^{(1)}$ dan $s_i^{(2)}$ adalah dua sinyal yang akan dibandingkan. Secara khusus, untuk mengukur hilangnya informasi dari sinyal yang direkonstruksi sehubungan dengan sinyal ideal (tidak terpengaruh oleh noise), dalam MS E∗, $s_i^{(1)}$ dan $s_i^{(2)}$ diwakili oleh yang terkuantisasi dan yang asli. sampel de-noise.

MOEA diterapkan pada sekumpulan kecil (set pelatihan) sampel yang mewakili keseluruhan sinyal. Pertama, sampel dide-noise menggunakan beberapa teknik standar. Kemudian, MOEA dijalankan dengan menggunakan sampel asli dan de-noise. Pada akhir proses optimisasi, kami memperoleh keluarga solusi yang tidak didominasi sehubungan dengan dua tujuan. Encoder mengkodekan indeks yang mengidentifikasi sel dengan mengeksploitasi kamus yang dihasilkan dengan menggunakan algoritma Huffman. Algoritma ini menyediakan metode sistematis untuk merancang kode biner dengan panjang rata-rata terkecil yang mungkin untuk sekumpulan frekuensi kemunculan simbol tertentu. Untuk menentukan perkiraan frekuensi ini, kami mengeksploitasi lagi set pelatihan. Untuk quantizer tertentu, kami menghitung probabilitas yang terjadi pada setiap indeks kuantisasi ketika

mengkuantisasi perbedaan antara sampel berturut-turut dari set pelatihan. Setelah representasi kata kode biner dari setiap indeks kuantisasi telah dihitung dan disimpan dalam node sensor, fase pengkodean direduksi menjadi konsultasi tabel pencarian.

Terakhir, untuk menilai kinerja algoritme kompresi menggunakan quantizer khusus, kami menggunakan rasio kompresi CR yang didefinisikan sebagai:

$$CR = \left(1 - \frac{comSize}{origSize}\right) \times 100\%, \qquad (8.3)$$

di mana *comSize* dan *origSize* masing-masing mewakili ukuran aliran bit terkompresi dan asli.

Pengodean Kromosom dan Operator Perkawinan

Setiap kromosom terdiri dari 2 × *LMAX* + 1 gen integer. Gen pertama (N) menetapkan jumlah sel yang akan dipertimbangkan dalam domain positif (kami memutuskan untuk mempertimbangkan quantizer dengan karakteristik simetris sehubungan dengan asal sumbu). Gen N dapat mengasumsikan nilai dalam [1, *LMAX*], di mana *LMAX* diatur oleh pengguna dan mengidentifikasi jumlah sel maksimum yang mungkin. Kami memperkenalkan *LMAX* untuk membatasi ruang pencarian. Di sisi lain, karena proses kuantisasi diterapkan pada perbedaan antara sampel berturut-turut dari sinyal lingkungan dan perbedaan ini biasanya tidak terlalu tinggi, untuk memperbaiki batas jumlah sel mempercepat pelaksanaan *MOEA* tanpa mempengaruhi kebaikan hasil akhir. Gen kedua (DZ) mewakili jarak antara level 0 dan batas atas a0 dari sel nol. Ketika kinerja tingkat-distorsi yang baik diminta ke quantizer, lebar sel-nol biasanya diperlakukan secara individual. Karena setiap input dalam sel nol dikuantisasi menjadi 0, sel ini sering disebut zona mati. *LMAX* 1 berikutnya memasangkan *Cl* gen mengkodifikasi, untuk setiap sel *S l*, jarak antara level *yl* dan ambang bawah *al−1 + 1* sel1, dan jarak antara level *yl* dan ambang atas *al* sel. Gen terakhir SR mewakili jarak antara level yLMAX dan ambang bawah aLMAX −1 + 1 dari sel *S – LMAX*. Setiap nilai input yang lebih besar dari ambang *aLMAX −1 + 1* sel terakhir dikuantisasi ke level terakhir (wilayah saturasi). Oleh karena itu, kecuali untuk gen pertama, gen yang tersisa menggambarkan sel Sl dengan menentukan posisi level dan posisi ambang bawah dan atas [*al−1 + 1, al*]. Semua gen, kecuali gen N, mengasumsikan nilai dalam [0, D/4], di mana D adalah domain perbedaan.

Sebagai operator kawin, kami telah menerapkan operator crossover satu titik klasik dan operator mutasi gen. Titik persekutuan dari persilangan satu titik dipilih dengan mengekstraksi angka secara acak dalam (*1, 2 LMAX + 1*). Operator crossover diterapkan dengan probabilitas PX; operator mutasi diterapkan setiap kali crossover tidak diterapkan. Setelah menerapkan operator kawin apa pun, kami melakukan pemeriksaan berikut: kami menghitung jumlah *NA* sel yang sebenarnya diaktifkan oleh set pelatihan. Jika *N > NA*, maka kita atur *N* menjadi *NA*. Dengan demikian, kami membuat kromosom konsisten dengan perhitungan entropi (memang, sel, yang diaktifkan tanpa sampel, memiliki entropi sama dengan 0 dan karena itu tidak mempengaruhi entropi).

8.4 LOKALISASI NODE DI WSN: SOLUSI BERBASIS MOEA

Pada bagian ini kami menjelaskan pendekatan berbasis MOEA kami untuk memecahkan masalah lokalisasi berbutir halus di WSN. Secara khusus, Bagian ini

memperkenalkan pernyataan masalah. Pada Bagian ini juga penulis menunjukkan ikhtisar pendekatan kami dengan membahas kendala berbasis topologi dan tujuan yang digunakan untuk mengevaluasi kebaikan estimasi topologi. Terakhir, Bagian terakhis sub bab ini menjelaskan representasi kromosom dan operator kawin.

Rumusan Masalah

Mari kita anggap bahwa WSN memiliki n node yang dikerahkan di T = [0, 1] × [0, 1] ⊂ R2 dan bahwa node 1 sampai m, dengan m < n, adalah node jangkar yang koordinatnya **pi** = *(xi, yi)* ∈ *T*, i = 1,..., m, diketahui. Selanjutnya, kita asumsikan bahwa dua simpul, katakanlah i dan j, dapat berkomunikasi satu sama lain jika dan hanya jika *rij* ≤ *R*, di mana rij = ‖**p**i – **p**j‖ adalah jarak sebenarnya antara simpul *i* dan *j* (‖–‖ menunjukkan norma Euclidean) dan *R* adalah radius konektivitas (asumsi ini, dilambangkan sebagai model disk, sering dibuat dalam literatur, meskipun hasilnya merupakan perkiraan kasar dari kenyataan). Akhirnya, kita anggap bahwa jika simpul *i* dan *j* berada dalam jangkauan konektivitas satu sama lain, maka jarak antar simpul dij dapat diperkirakan dengan menggunakan beberapa teknik pengukuran (lihat Bagian 8.1) dan dapat dimodelkan sebagai

$$d_{ij} = r_{ij} + e_{ij} \tag{8.4}$$

kami berasumsi bahwa kesalahan pengukuran *eij* mengikuti distribusi Gaussian nol-mean dengan varians σ^2, dan variabel acak *eij* dan *ekl* secara statistik independen untuk *(i, j) ≠ (k, l).* Kami merujuk ke simpul *j* sehingga *rij* ≤ *R* sebagai tetangga tingkat pertama dari simpul *i*. Membiarkan menjadi himpunan tetangga tingkat pertama dari simpul i dan komplemennya.

$$N_i = \{j \in 1 \ldots n, j \neq i : r_{ij} \leq R\} \tag{8.5}$$

$$\overline{N}_i = \{j \in 1 \ldots n, j \neq i : r_{ij} > R\} \tag{8.6}$$

Kita asumsikan bahwa himpunan *Ni* dan $\overline{Ni}$ diketahui untuk semua *i = 1, ..., n*. Ini adalah asumsi yang masuk akal, karena setiap node dapat dengan mudah menentukan node lain mana yang dapat berkomunikasi dengannya. Mengingat posisi m node jangkar dan perkiraan jarak antar node *dij*, kami bertujuan untuk menentukan posisi node non-jangkar dengan mengeksploitasi pendekatan berbasis MOEA.

Gambaran Umum Pendekatan Kami

Berdasarkan posisi node jangkar dan rentang konektivitas, kita dapat mengklasifikasikan node non-jangkar i menjadi tiga kelas yang berbeda:

- Kelas 1: setidaknya sebuah simpul jangkar adalah tetangga tingkat pertama dari i
- Kelas 2: i bukan milik Kelas 1 dan setidaknya simpul non-jangkar, yang memiliki simpul jangkar sebagai tetangga tingkat pertama, adalah tetangga tingkat pertama dari i
- Kelas 3: i bukan milik Kelas 1 maupun Kelas 2.

Keanggotaan node non-anchor ke salah satu dari tiga kelas memungkinkan membatasi ruang di mana node dapat ditemukan. Informasi ini dapat dimanfaatkan baik dalam pembangkitan populasi awal maupun dalam pelaksanaan operator kawin sehingga dapat mempercepat pelaksanaan MOEA dengan menghindari pembangkitan solusi yang tentunya tidak dapat optimal.

Setiap estimasi dikaitkan dengan vektor dari dua elemen, yang mewakili nilai dari dua fungsi tujuan *CF* dan *CV*. Misalkan *pˆi = (xˆi, yˆi), i = m + 1,..., n* adalah perkiraan posisi node non-anchor i.

$$CF = \sum_{i=m+1}^{n} \left(\sum_{j \in N_i} \left(\hat{d}_{ij} - d_{ij} \right)^2 \right), \tag{8.7}$$

di mana *dij* dan *dˆij* merepresentasikan jarak terukur dan estimasi antara node *i* dan *j*.
CV menghitung jumlah kendala konektivitas yang tidak dipenuhi oleh geometri kandidat, dan didefinisikan sebagai

$$CV = \sum_{i=1}^{n} \left(\sum_{j \in N_i} \delta_{ij} + \sum_{j \in \overline{N}_i} \left(1 - \delta_{ij} \right) \right), \tag{8.8}$$

di mana δij = 1 jika dˆij > R, dan 0 sebaliknya.

Untuk mengevaluasi keakuratan perkiraan, kami mempertimbangkan kesalahan pelokalan yang dinormalisasi (NLE), yang didefinisikan sebagai

$$NLE = \frac{1}{R} \sqrt{\frac{1}{(n-m)} \sum_{i=m+1}^{n} \left(\|\mathbf{p}_i - \hat{\mathbf{p}}_i\|^2 \right)} \times 100\%. \tag{8.9}$$

Dengan demikian, dengan asumsi estimasi tidak bias, *NLE* dapat diartikan sebagai rasio standar deviasi terhadap radius konektivitas.

Pengodean Kromosom dan Operator Perkawinan

Setiap kromosom mengkodekan posisi semua node non-anchor dalam jaringan. Dengan demikian, setiap kromosom terdiri dari *n – m* pasangan bilangan real, di mana setiap pasangan mewakili koordinat *xˆ* dan *yˆ* dari node non-anchor. Kisaran variasi setiap koordinat dibatasi oleh kendala geometris yang telah dijelaskan sebelumnya. Kami menegakkan kepatuhan dengan kendala ini pada populasi awal. Selanjutnya, kapan pun mutasi diterapkan selama proses evolusi, hanya individu bermutasi yang memenuhi batasan ini yang dihasilkan.

Operator mutasi pertama, dilambangkan dengan operator mutasi node, melakukan mutasi seperti seragam: posisi setiap node sensor non-anchor dimutasi dengan probabilitas *PU = 1/(n – m).* Posisi dihasilkan secara acak dalam batasan geometris yang dikenakan pada lokasi node tertentu. Operator mutasi kedua, yang disebut operator mutasi ketetanggaan, diterapkan ketika operator pertama tidak dipilih. Operator mutasi lingkungan bermutasi, dengan probabilitas *PU*, posisi setiap node sensor non-anchor dalam batasan geometris yang ditentukan untuk node tertentu, tetapi tidak seperti operator pertama, ini menerapkan terjemahan kaku yang sama, yang telah membawa node yang bermutasi. *i* dari posisi pra-mutasi ke posisi pasca-mutasi, ke tetangga *i* dengan probabilitas tertentu (probabilitas translasi kaku).

8.5 ALGORITMA EVOLUSI MULTI-TUJUAN

Optimalisasi evolusioner multi-objektif telah diselidiki oleh beberapa penulis dalam beberapa tahun terakhir dan beberapa algoritma yang berbeda telah diusulkan. Beberapa yang paling populer di antara algoritma ini adalah *Strength Pareto Evolutionary Algorithm* (SPEA) dan evolusinya (SPEA2), *Niched Pareto Genetic Algorithm* (NPGA), versi yang berbeda dari *Strategi Pareto Archived Evolution* (PAES), dan *Non-dominated Sorting Genetic Algorithm* (NSGA) dan evolusinya (NSGA-II). Berikut ini, kami secara singkat memperkenalkan NSGA-II dan PAES, yang telah terbukti sangat efektif dalam menghasilkan kompresor data yang kehilangan kesadaran energi dan masing-masing memecahkan masalah lokalisasi node.

NSGA–II

NSGA-II adalah algoritma genetika multi-objektif berbasis populasi. Setiap individu populasi diasosiasikan dengan peringkat yang sama dengan tingkat non-dominansinya (1 untuk tingkat terbaik, 2 untuk tingkat berikutnya-terbaik, dan seterusnya). Untuk menentukan tingkat non-dominan (dan konsekuensinya peringkat), untuk setiap individu *p,* jumlah *np* individu yang mendominasi *p* dan himpunan *S p* individu yang didominasi oleh *p* dihitung. Semua individu dengan *np = 0* termasuk dalam tingkat non-dominan terbaik yang terkait dengan peringkat 1. Untuk menentukan individu yang terkait dengan peringkat 2, untuk setiap solusi *p* dengan peringkat 1, setiap anggota *q* dari himpunan *S p* dikunjungi dan nq dikurangi dengan satu. Jika *nq* menjadi nol, maka *q* milik tingkat non-dominan yang terkait dengan peringkat 2. Prosedur ini diulangi untuk setiap solusi dengan peringkat 2, peringkat 3 dan seterusnya hingga semua front teridentifikasi. NSGA–II dimulai dari populasi acak awal *P0* dari individu N*pop* yang diurutkan berdasarkan non-dominan. Pada setiap iterasi *t,* populasi keturunan *Qt* dengan ukuran *Npop* dihasilkan dengan memilih individu yang kawin melalui seleksi turnamen biner, dan dengan menerapkan operator persilangan dan mutasi.

Populasi induk *Pt* dan populasi keturunan *Qt* digabungkan sehingga menghasilkan populasi baru *Pext = Pt ∪ Qt*. Kemudian, peringkat ditetapkan untuk setiap individu di *Pext*.

Akhirnya, *Pext* dibagi menjadi beberapa front berbeda yang tidak didominasi, satu untuk setiap peringkat yang berbeda.

Dalam setiap front, ukuran crowding tertentu, yang mewakili jumlah jarak ke individu terdekat di sepanjang setiap tujuan, digunakan untuk menentukan urutan di antara individu: untuk menutupi keseluruhan ruang tujuan, individu dengan jarak crowding yang besar lebih disukai untuk individu dengan jarak kerumunan kecil. Populasi induk baru *Pt+1* dihasilkan dengan memilih individu *Npop* terbaik (pertama-tama dengan mempertimbangkan pengurutan di antara front dan kemudian di antara individu) dari *Pext*.

PAES

PAES adalah MOEA berbasis arsip. Itu mempertahankan solusi arus tunggal *c,* dan, pada setiap iterasi, menghasilkan solusi baru *m* dari *c*, dengan hanya menggunakan operator mutasi. Kemudian, *m* dibandingkan dengan *c.* Tiga kasus berbeda dapat muncul:

I. *c* mendominasi *m: m* dibuang;

II. *m* mendominasi *c: m* dimasukkan ke dalam arsip dan kemungkinan solusi dalam arsip yang didominasi oleh m dihapus; *m* menggantikan *c* dalam peran solusi saat ini;

III. tidak ada kondisi yang terpenuhi: *m* ditambahkan ke arsip hanya jika didominasi oleh tidak ada solusi yang terkandung dalam arsip; m menggantikan *c* dalam peran solusi saat ini hanya jika *m* milik suatu daerah dengan derajat kepadatan lebih kecil dari, atau sama dengan, daerah *c.*

Tingkat kepadatan dihitung dengan terlebih dahulu membagi ruang di mana solusi arsip terletak pada sejumlah (numReg) dari wilayah berukuran sama dan kemudian dengan menghitung solusi yang dimiliki oleh wilayah tersebut. Banyaknya solusi ini menentukan tingkat kepadatan suatu wilayah. Pendekatan ini cenderung memilih solusi yang dimiliki oleh daerah yang kurang padat, untuk menjamin distribusi solusi yang seragam di sepanjang front Pareto.

PAES berakhir setelah jumlah evaluasi maksimum yang diberikan. Strategi penerimaan solusi kandidat menghasilkan arsip yang hanya berisi solusi yang tidak didominasi. Pada penghentian PAES, arsip (paling banyak archS ize) menyertakan kumpulan solusi yang merupakan perkiraan dari front Pareto. Pada awalnya, arsip kosong dan solusi arus pertama dihasilkan secara acak.

8.6 HASIL EKSPERIMEN UNTUK PENDEKATAN KOMPRESI DATA

Di bagian ini, pertama kami memperkenalkan pengaturan eksperimental yang digunakan dalam percobaan. Kemudian, kami menunjukkan hasil yang diperoleh dengan menerapkan MOEA yang berbeda untuk menghasilkan kompresor data yang sadar energi dengan pertukaran yang berbeda antara *MS E∗* dan entropi *H* pada set pelatihan yang diberikan. Dengan menganalisis secara statistik distribusi hipervolume yang diperoleh oleh MOEA yang berbeda, kami menunjukkan bahwa NSGA–II mengungguli MOEA lainnya. Jadi, kami memilih NSGA–II sebagai MOEA untuk pendekatan kami dan menerapkannya ke kumpulan data lain yang berbeda dari kumpulan pelatihan. Akhirnya, kami membandingkan hasil yang diperoleh oleh NSGA-II dengan yang dicapai oleh algoritma canggih, yaitu LTC, yang baru-baru ini diusulkan dalam literatur. Kami akan menunjukkan bahwa pendekatan kami dapat mencapai rasio kompresi yang signifikan meskipun kesalahan rekonstruksi dapat diabaikan dan mengungguli LTC dalam hal tingkat kompresi dan kompleksitas.

Penyiapan Eksperimental

Demi singkatnya, kami akan membahas penerapan pendekatan kami hanya untuk sampel yang dikumpulkan oleh sensor suhu. Kami menggunakan data publik dari penyebaran SensorScope. Kami memilih untuk mengadopsi kumpulan data domain publik daripada menghasilkan data sendiri untuk membuat penilaian seadil mungkin. WSN yang diadopsi dalam penerapan menggunakan tipe node TinyNode, yang menggunakan mikrokontroler TI MSP430, radio Xemics XE1205 dan modul sensor Sensirion SHT75. Modul ini adalah chip tunggal yang mencakup sensor suhu celah pita, digabungkan ke ADC 14-bit dan sirkuit antarmuka serial. Sensirion SHT75 dapat mendeteksi suhu udara pada kisaran [-20^o C, +60^o C]. Setiap keluaran ADC *raw_t* direpresentasikan dengan resolusi 14bit dan biasanya diubah menjadi ukuran *t* dalam derajat Celsius (^{o}C). Kumpulan data yang dikumpulkan dalam penerapan berisi langkah-langkah *t.* Di sisi lain, algoritma kami bekerja pada data *t* mentah. Jadi, sebelum menerapkan algoritma, kami mengekstrak *t* mentah dari *t.*

Karena pendekatan kami memerlukan set pelatihan untuk eksekusi MOEA, kami memilih node, yaitu node 101, dari penyebaran Fish-Net (salah satu penyebaran SensorScope), dan menggunakan data sampel suhu N = 5040 yang dikumpulkan dari bulan Agustus. 9 2007 hingga 16 Agustus 2007 (kami akan merujuk dataset ini dengan nama simbolis FN). Karena sampel dikumpulkan pada frekuensi 1 sampel setiap 2 menit, set pelatihan sesuai dengan pemantauan seminggu. Bagian yang diekstraksi dari sinyal asli pertama-tama dihilangkan deraunya, menggunakan penyusutan wavelet dan metode thresholding. Secara khusus, kami telah menggunakan wavelet Symmlet 8, tingkat dekomposisi sama dengan 5 dan aturan ambang batas universal lunak untuk ambang batas koefisien detail pada setiap tingkat. Proses de-noising telah dilakukan dengan menggunakan fungsi bawaan Matlab standar.

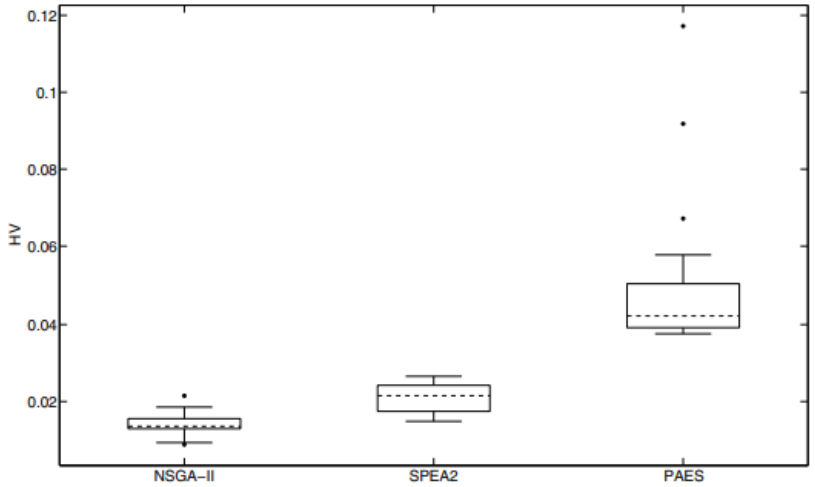
Memilih MOEA untuk Masalah Spesifik

Sebagai langkah awal, untuk memilih MOEA terbaik untuk mengatasi masalah tertentu, kami telah mengeksekusi 50 proses independen NSGA–II, SPEA2 dan PAES dan telah membandingkan ketiga MOEA dengan menggunakan hypervolume.

Kami telah menetapkan LMAX = 16 dan jumlah maksimum TMAX evaluasi menjadi 106. Selanjutnya, ketika mutasi diterapkan, gen N bermutasi dengan probabilitas 0,5. Jika gen N tidak bermutasi, maka salah satu gen yang tersisa dipilih secara acak. Untuk NSGA–II dan SPEA2 kami mengadopsi ukuran populasi *Npop* = 100 dan probabilitas persilangan *PX* = 0,8. Untuk PAES, kami menggunakan sejumlah wilayah *numReg* = 5 dan ukuran arsip *archS ize* = 100.

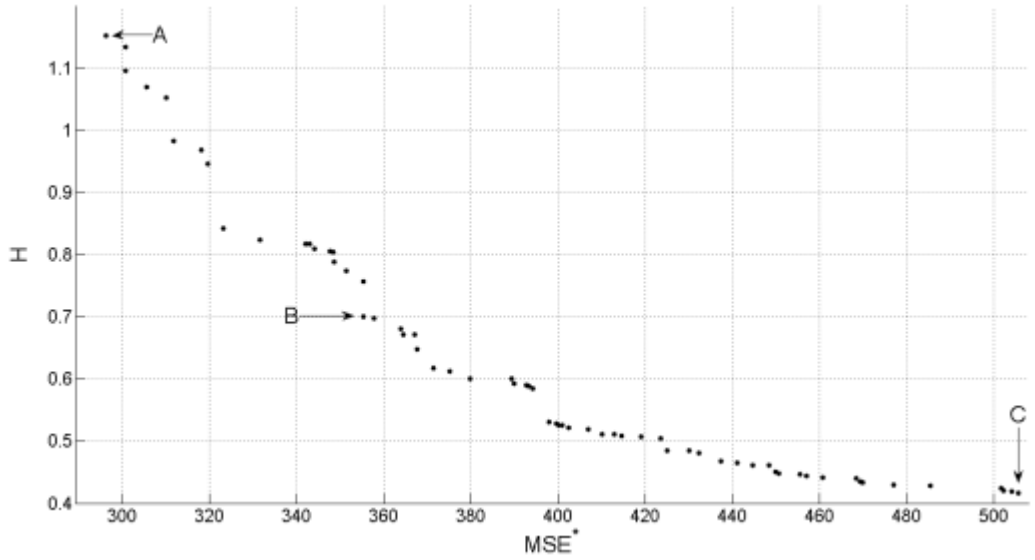
Dalam masalah optimisasi dua tujuan, hypervolume mengukur luas porsi ruang objektif yang didominasi lemah oleh pendekatan depan Pareto. Untuk membandingkan ketiga MOEA kami telah menggunakan indikator hypervolume (HV) yang dihitung dalam paket penilaian kinerja yang disediakan dalam perangkat PISA. Di sini, semakin rendah nilai HV, semakin tinggi kualitas algoritma yang sesuai.

Gambar 8.2 menunjukkan plot kotak indikator HV untuk ketiga MOEA. Dari analisis tiga kotak-plot kita dapat menyimpulkan bahwa NSGA-II mengungguli dua MOEA lainnya dalam masalah tertentu.



Gambar 8.2. Plot kotak indikator hypervolume HV untuk NSGA–II, SPEA2 dan PAES

Untuk mengevaluasi stabilitas NSGA–II dalam masalah spesifik, pada penelitian sebelumnya juga telah melakukan analisis distribusi front yang dihasilkan dalam 50 percobaan pada bidang $MSE * H$. Kami telah mengamati bahwa front yang dihasilkan dalam uji coba yang berbeda cukup dekat satu sama lain, sehingga menegaskan bahwa algoritme tersebut cukup stabil. Pada Gambar 8.3 kami menunjukkan front Pareto akhir yang diperoleh dari salah satu percobaan. Kita dapat mengamati bahwa bagian depannya cukup lebar dan solusinya dicirikan oleh trade-off yang baik antara *H* dan $MSE*$.



Gambar 8.3. Contoh pendekatan pareto depan

Hasil Percobaan

Kami telah menguji pendekatan kami dengan NSGA–II dengan menggunakan tiga set data suhu dari dua penerapan SensorScope, yaitu penerapan Grand–St–Bernard dan Le Genepi. Tabel 8.1 merangkum karakteristik utama dari kumpulan data (untuk kelengkapan, kami telah menunjukkan dalam tabel juga karakteristik dari kumpulan pelatihan). Berikut ini kami akan mengacu pada kumpulan data tersebut dengan menggunakan nama simboliknya. Secara khusus, kami telah menghitung kisaran suhu *([min, maks]),* entropi informasi H dari sampel dan *MS EREF,* yang mewakili *MS E* antara kumpulan data asli dan versi de-noise-nya. Dengan menganalisis kisaran suhu dari berbagai sensor, kami menyadari bahwa sensor mengukur suhu yang sangat dingin dan hangat. Kami secara tegas memilih kumpulan data ini untuk menunjukkan bahwa pendekatan kami tidak bergantung pada nilai suhu tertentu yang diukur oleh sensor.

Tabel 8.1. Karakteristik utama dari kumpulan data

Nama Simbol	Nama Depatemen	ID Node	Jumlah Sampel	Waktu Interfal (bb/hh/tttt)		[min, max]	H	MS EREF
				dari	ke			
FN	Fis_Net	101	5040	08/09/2007	08/16/2007	[7,7,26,51]	10.01	349.51
GS B1	GR.St Bernard	31	2160	10/04/2007	10/062007	[4,16,14,32]	8.95	1417.83
GS B2	GR.St Bernard	31	2160	10/20/2007	10/22/2007	[-11,0,75]	9.15	1615.99
LG	Le Genepi	2	4320	09/25/2007	10/30/2007	[-4,66,8,89]	9.42	1251.04

Untuk melakukan analisis yang akurat dari beberapa solusi, kami memilih dari depan pada Gambar. 8.3 tiga quantizer signifikan: solusi (A) dan (C) ditandai dengan, masing-masing, H tertinggi dan MS E∗, dan solusi (B) ditandai dengan pertukaran yang baik antara H dan MS E∗. Solusi (A), (B) dan (C) sesuai dengan aturan kuantisasi yang ditunjukkan pada Tabel 8.2. Di sini, maxD menunjukkan nilai maksimum yang mungkin dari perbedaan di dalam domain D.

Tabel 8.2. Aturan kuantisasi untuk solusi (A), (B) dan (C)

1 cell		solution (A)	solution (B)	solution (C)
0	S_0	$[0,11]$	$[0,29]$	$[0,32]$
	y_0	0	0	0
1	S_1	$[12,\max_D]$	$[30,62]$	$[33,80]$
	y_1	12	32	49
2	S_2	–	$[63,\max_D]$	$[81,87]$
	y_2	–	71	85
3	S_3	–	–	$[88,\max_D]$
	y_3	–	–	120

Tabel 8.3 menunjukkan nilai *CR, MS E∗, MS EN* dan *MS ED* yang diperoleh dari tiga quantizer terpilih pada dataset *FN, GS B1, GS B2* dan *LG. MS EN* dan *MS ED*, masing-masing, *MS E* dihitung antara noise yang direkonstruksi dan sinyal asli noise, dan antara sinyal asli yang direkonstruksi dan dihilangkan noise.

Kita dapat mengamati bahwa semua solusi mencapai pertukaran yang baik antara rasio kompresi dan MS Es. Selanjutnya, tidak ada perbedaan yang cukup besar antara hasil yang diperoleh dalam set pelatihan (FN) dan yang dicapai dalam tiga set data lainnya. Hasil ini bisa dibilang cukup mengejutkan. Memang, kami menyoroti bahwa optimasi dan algoritme manusia dieksekusi menggunakan sinyal yang dikumpulkan oleh node dalam penyebaran yang berbeda (yaitu, FN). Dengan demikian, dataset GS B1, GS B2 dan LG sama sekali tidak diketahui skema kompresinya. Kami sadar bahwa prosedur yang diadopsi untuk set pelatihan dapat diterapkan secara mendalam juga ke set data lainnya, untuk menemukan solusi ad-hoc untuk penerapan tertentu. Di sisi lain, ketiga dataset suhu dikumpulkan, meskipun di tempat dan waktu yang berbeda, oleh node sensor yang sama dengan jenis sensor suhu yang sama dan frekuensi pengambilan sampel yang sama.

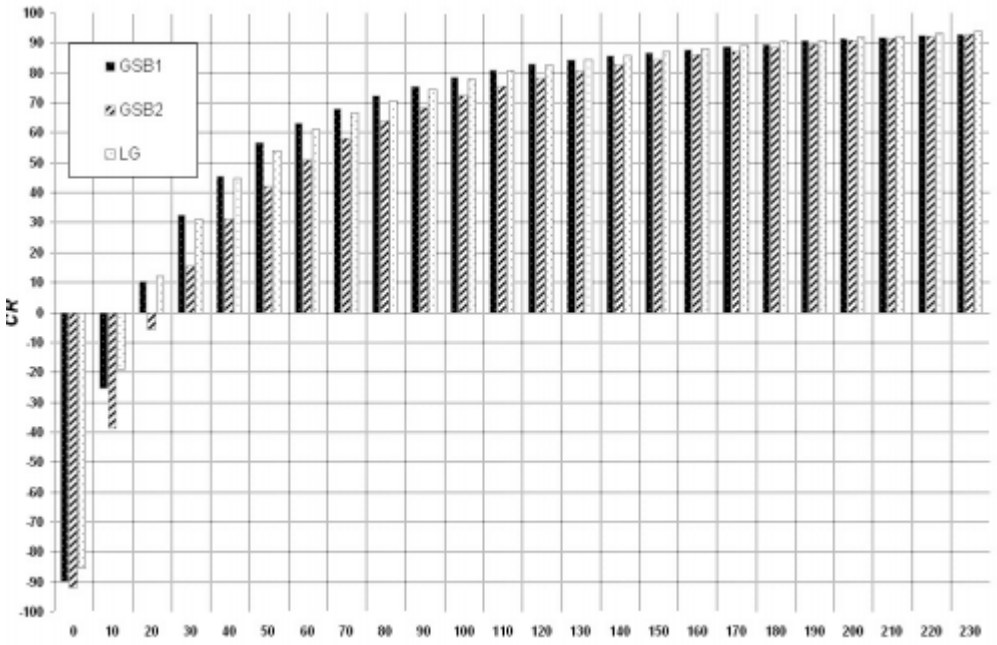
Untuk menilai keefektifan solusi yang dihasilkan oleh NSGA-II sehubungan dengan kompresor yang ditentukan secara sengaja yang baru-baru ini diusulkan dalam literatur, kami memilih solusi (B), yang mewakili pertukaran yang baik antara rasio kompresi dan MS E∗, dan membandingkan kinerjanya dengan algoritma LTC.

Tabel 8.3. Hasil yang diperoleh solusi (A), (B) dan (C) pada keempat dataset

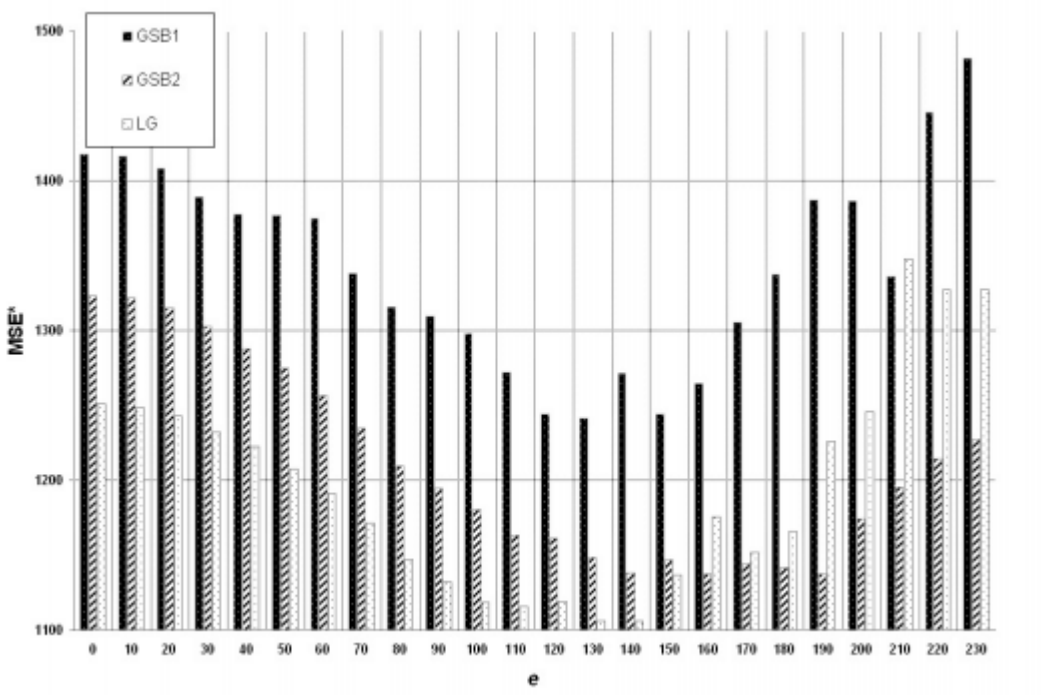
Dataset	Solution	CR	MSE^*	MSE_N	MSE_D
FN	A	91.94	296.20	141.52	49.19
	B	92.90	397.89	193.37	81.00
	C	93.12	505.69	232.92	63.07
$GSB1$	A	90.69	685.38	799.64	106.33
	B	91.41	1286.82	252.69	30.56
	C	91.76	1467.10	269.77	36.94
$GSB2$	A	89.78	863.21	917.18	112.40
	B	90.05	1350.20	252.70	20.89
	C	90.25	1685.92	253.25	19.16
LG	A	90.87	473.70	909.27	99.36
	B	91.46	1027.95	292.08	32.52
	C	91.50	1320.84	255.50	36.38

Perbandingan dengan LTC

LTC menghasilkan satu set segmen garis yang membentuk fungsi kontinu sepotong-sepotong. Fungsi ini mendekati dataset asli sedemikian rupa sehingga tidak ada sampel asli yang lebih jauh dari kesalahan tetap e dari segmen garis terdekat. Jadi, sebelum mengeksekusi algoritma LTC, kita harus mengatur error e. Kami memilih e sebagai persentase dari Sensor Manufactured Error (S ME). Dari lembar data sensor Sensirion SHT75, didapatkan S ME = 0.3ºC untuk temperatur. Untuk menganalisis tren CR sehubungan dengan peningkatan nilai e, kami memvariasikan *e* dari 0% menjadi 230% UKM dengan langkah 10%.



Gambar 8.4. Rasio kompresi yang diperoleh algoritma LTC untuk nilai error yang berbeda pada ketiga dataset



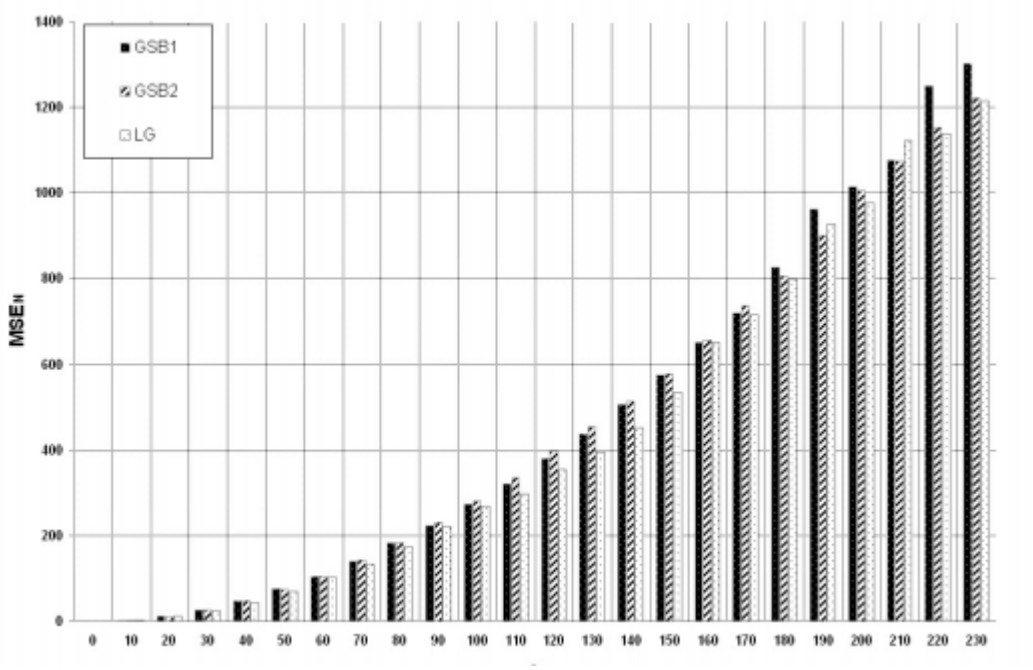
Gambar 8.5. MS E∗s diperoleh dengan algoritma LTC untuk nilai error yang berbeda pada ketiga dataset

Gambar. 8.4–8.7 menunjukkan, untuk ketiga set data, tren *CR, MS E∗, MS EN* dan *MS ED*, masing-masing, diperoleh oleh algoritma LTC untuk nilai kesalahan yang berbeda. Karena pendekatan tertentu berdasarkan perkiraan linier yang digunakan dalam LTC, kami mengamati bahwa versi LTC lossless (sesuai dengan e = 0) menghasilkan CR negatif, yaitu ukuran data terkompresi lebih besar dari data asli. Selanjutnya, kami mencatat bahwa CR yang menarik diperoleh hanya meskipun ada kesalahan kompresi yang relevan. Dengan membandingkan Tabel 8.3, di mana kami telah menunjukkan CR yang diperoleh oleh algoritme kami, dengan Gambar. 8.4–8.7, kami dapat mengamati bahwa algoritme LTC dapat mencapai CR yang sama dengan algoritma kami, tetapi meskipun MS Es cukup tinggi. Misalnya, algoritma kami mencapai CR yang setara dengan 91,41 untuk GS B1.

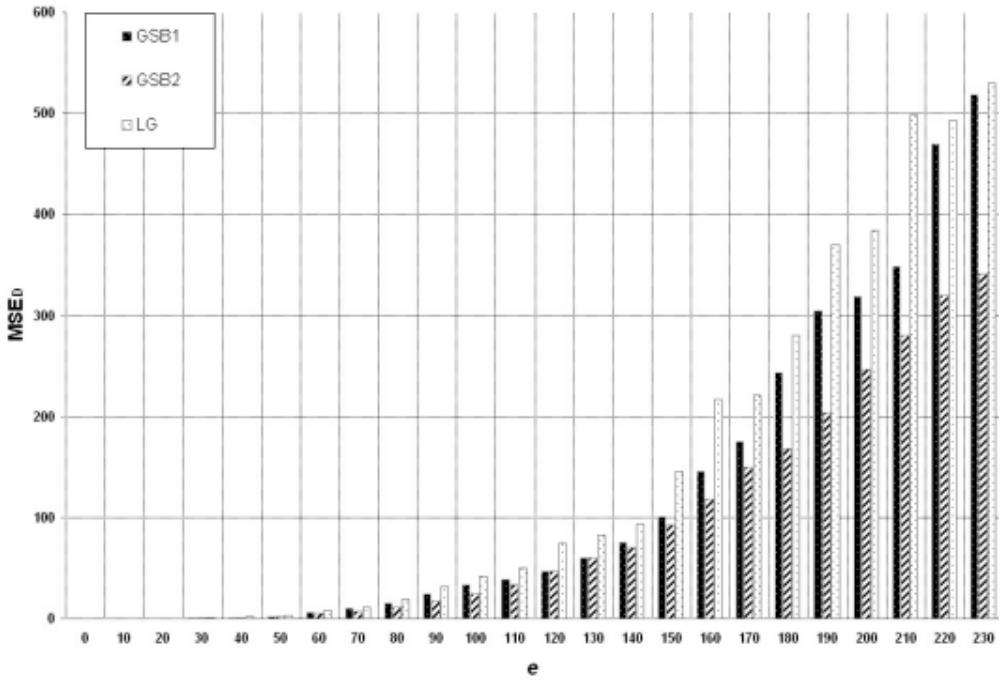
Dari Gambar 8.4, kita dapat mengamati bahwa, untuk mencapai CR yang serupa, algoritma LTC harus dijalankan dengan nilai e antara 200% dan 210% dari SME. Dari Gambar 8.5, kita dapat menyimpulkan bahwa nilai e lead ini memiliki MS E∗ antara 1386.46 dan 1335.80 melawan MS E∗ = 1286.81 untuk algoritma kita. Pertimbangan serupa dapat dibuat pada MS EN dan MS ED dan Gambar. 8.6 dan Gambar. 8.7. Tabel 8.4 menunjukkan korespondensi antara CR yang dicapai oleh algoritme kami, dan CR, MS E∗, MS EN, dan MS ED yang diperoleh LTC pada tiga set data. Dengan membandingkan Tabel 8.4 dengan Tabel 8.3, kami dapat mengamati bahwa algoritme kami memperoleh MS Es yang lebih rendah daripada LTC sesuai dengan CR yang sama (kecuali untuk GS B2 dan MS E∗). Misalnya, untuk GS B1 dan CR sama dengan 91,41, MS EN (MS ED) berada di antara 1015,92 (318,83) dan 1076,30 (348,43) untuk LTC, sedangkan MS EN = 252,69 (MS ED = 30,56) untuk algoritme kami.

Tabel 8.4. Korespondensi antara CR yang dicapai oleh algoritme kami, dan CR dan MS Es yang dicapai oleh LTC pada tiga set data

Dataset	*CR*	*e* [*min*, *max*]	*CR* [*min*, *max*]	MSE^* [*min*, *max*]	MSE_N [*min*, *max*]	MSE_D [*min*, *max*]
*GSB*1	91.41	[200, 210]	[91.39, 91.94]	[1386.46, 1335.80]	[1015.92, 1076.30]	[318.83, 348.43]
*GSB*2	90.05	[190, 200]	[89.47, 90.68]	[1137.23, 1173.90]	[899.90, 1005.65]	[203.62, 246.59]
LG	91.46	[190, 200]	[90.88, 91.81]	[1225.95, 1245.53]	[926.28, 976.78]	[370.01, 384.54]



Gambar 8.6. MS EN diperoleh dengan algoritma LTC untuk nilai error yang berbeda pada ketiga dataset



Gambar 8.7. MS ED diperoleh dengan algoritma LTC untuk nilai error yang berbeda pada ketiga dataset

Karena kompresor telah digunakan pada node dengan kemampuan komputasi dan memori yang berkurang, perbandingan tidak hanya dapat mempertimbangkan rasio kompresi dan MS E, tetapi juga harus mempertimbangkan kompleksitas. Untuk tujuan ini, kami telah melakukan analisis komparatif pada jumlah instruksi yang dibutuhkan oleh kompresor kami dan oleh LTC yang mengeksploitasi simulator Sim-It Arm. Sim–It Arm adalah simulator kumpulan instruksi yang menjalankan program ARM tingkat sistem dan tingkat pengguna. Untuk LTC, kami menetapkan e ke ekstrem kiri interval e pada Tabel 8.4 (ingat bahwa ekstrem kiri adalah kasus yang paling disukai untuk algoritme LTC). Kami telah mengukur bahwa, rata-rata, algoritme kami mengeksekusi 6,06 instruksi untuk setiap bit yang disimpan terhadap 40,99 yang dieksekusi oleh LTC. Jadi, kami dapat menyimpulkan bahwa, meskipun algoritme kami mencapai MS Es yang lebih rendah pada bitrate yang sama dengan LTC, algoritma ini memerlukan jumlah instruksi yang lebih sedikit.

8.7 HASIL EKSPERIMEN UNTUK PENDEKATAN LOKALISASI NODE

Di bagian ini, pertama kami memperkenalkan pengaturan eksperimental yang digunakan dalam percobaan. Kemudian, kami menunjukkan hasil yang diperoleh dengan menerapkan PAES untuk memecahkan masalah lokalisasi simpul dan membandingkan hasil ini dengan hasil yang dicapai oleh dua algoritme canggih yang baru-baru ini diusulkan dalam literatur. Kami menyoroti bahwa, dalam semua percobaan, pendekatan kami mencapai akurasi yang cukup besar, sehingga menunjukkan keefektifan dan stabilitasnya, dan mengungguli rata-rata algoritme lainnya. PAES dipilih sebagai MOEA setelah percobaan panjang di mana kami membandingkan PAES dengan empat pendekatan evolusi multi-tujuan lainnya, termasuk NSGA-II.

Penyiapan Eksperimental

Penulis telah membangun topologi jaringan yang berbeda dengan secara acak menempatkan 200 node dengan distribusi yang seragam di *T.* Kami telah memvariasikan persentase simpul jangkar menjadi 8%, 10% dan 12% (sehingga setiap topologi terdiri dari 16, 20 dan 24 simpul jangkar dan masing-masing 184, 180 dan 176 simpul non-jangkar). Selanjutnya, kami memvariasikan radius konektivitas R dalam interval [0,11, 0,16] dengan langkah 0,01. Pengukuran jarak antara node tetangga dihasilkan sesuai dengan model (8.4). Kami berasumsi bahwa perkiraan jarak ini berasal dari pengukuran RSS, yang biasanya dipengaruhi oleh log-normal shadowing dengan standar deviasi dari kesalahan yang sebanding dengan rentang aktual *rij.* Dengan demikian, variansi eij diberikan oleh $\sigma 2 = \alpha 2r2$. Nilai $\alpha = 0,1$ digunakan dalam simulasi.

Untuk setiap nilai *R,* dihasilkan 10 topologi jaringan acak. Setelah langkah ini, skenario yang berbeda dibangun dengan memvariasikan persentase node jangkar. Dengan demikian, kami dapat memperoleh kontrol yang lebih baik pada efek dari jari-jari konektivitas yang berbeda dan persentase node jangkar yang berbeda pada kesalahan lokalisasi yang dinormalisasi. Tabel 8.5 menunjukkan nilai rata-rata beberapa indikator jaringan, yaitu jumlah node tetangga anchor dan non-anchor level pertama, persentase node anchor, persentase node non-anchor yang diklasifikasikan dalam Kelas 1 dan Kelas 2 (nilai persentase simpul di Kelas 3 dapat dengan mudah disimpulkan dari dua yang pertama), persentase simpul non-

jangkar tanpa simpul jangkar di lingkungannya dan persentase simpul non-jangkar yang memiliki setidaknya 3 simpul jangkar di lingkungannya. Analisis Tabel 8.5 mengungkapkan bahwa, ketika radius konektivitas dan persentase node jangkar rendah, masalah lokalisasi menjadi sangat kompleks: memang, dengan radius konektivitas R = 0,11 dan 8% node jangkar, hanya sedikit sebagian kecil node non-anchor dapat mengandalkan 3 atau lebih tetangga anchor (2,28%), sementara lebih dari setengahnya (57,72%) berkomunikasi tanpa node anchor. Selain itu perlu dicatat bahwa, bahkan ketika radius konektivitas meningkat menjadi 0,16 dan persentase simpul jangkar menjadi 12%, persentase rata-rata simpul non-jangkar tanpa tetangga jangkar tidak dapat diabaikan (18,64%), sedangkan persentase rata-rata node non-anchor dengan 3 atau lebih tetangga anchor masih sangat rendah (24,55%).

Tabel 8.5. Nilai rata-rata dari beberapa indikator jaringan utama untuk jari-jari konektivitas yang berbeda dan persentase node jangkar

R	number of first–level neighbors	anchor (%)	Class 1 (%)	Class 2 (%)	0 anchor (%)	3(or more) anchors (%)
0.11	6.86	8	42.28	31.68	57.72	2.28
		10	50.44	32.72	49.56	3.56
		12	56.02	30.68	43.98	5.97
0.12	8.09	8	51.03	31.85	48.97	2.12
		10	59.67	29.22	40.33	4.50
		12	66.48	25.80	33.52	5.85
0.13	9.41	8	58.04	32.72	41.96	2.17
		10	65.28	29.72	34.72	5.33
		12	69.72	26.14	30.28	9.43
0.14	10.84	8	56.52	30.43	43.48	5.98
		10	67.28	25.94	32.72	8.94
		12	75.06	20.68	24.94	14.20
0.15	12.28	8	62.01	29.35	37.99	7.66
		10	70.94	24.11	29.06	12.78
		12	75.74	21.42	24.26	17.16
0.16	14.16	8	67.83	26.20	32.17	11.09
		10	74.17	22.72	25.83	17.22
		12	81.36	16.76	18.64	24.55

Hasil Eksperimen dan Perbandingan

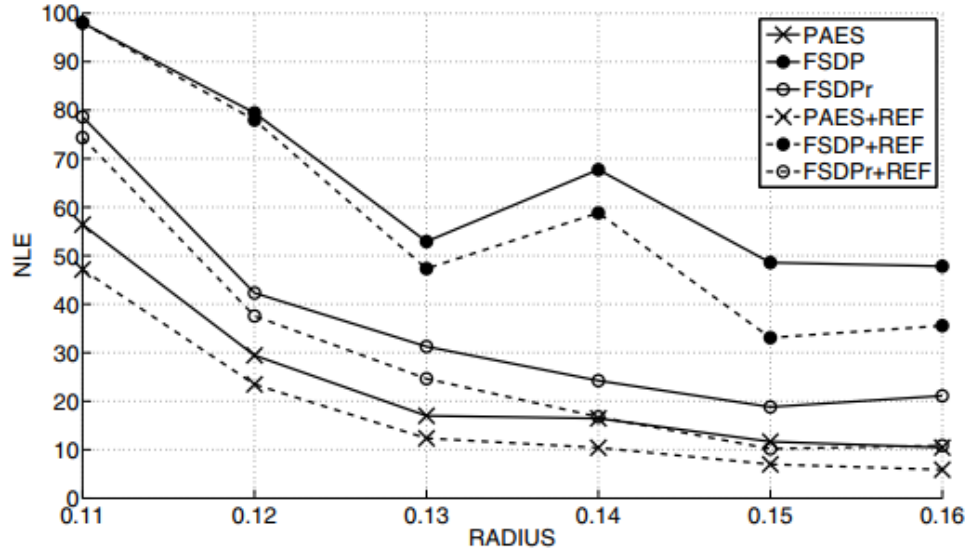
Untuk setiap skenario, 15 uji coba PAES dieksekusi. Kami menetapkan *archS ize* = 20, *numReg* = 5, *maxEvals* = 4 105. Operator mutasi pertama diterapkan dengan probabilitas 0,9 dan probabilitas translasi kaku diatur ke 0,3. Kami telah memverifikasi bahwa bagian depan dekat satu sama lain, sehingga menegaskan stabilitas pendekatan. Selanjutnya, kami telah menunjukkan dengan menggunakan uji Wilcoxon bahwa tidak ada perbedaan statistik dalam hal NLE di antara solusi dalam perkiraan akhir Pareto front. Dengan demikian, kita dapat memilih solusi apapun untuk melakukan perbandingan dengan teknik lokalisasi lainnya. Demi singkatnya, kami mempertimbangkan solusi unik, yaitu solusi yang ditandai dengan nilai CF

terendah, dan membandingkan kinerja yang diperoleh oleh solusi tersebut dalam topologi yang berbeda dengan pendekatan relaksasi cembung yang paling akurat, yaitu pendekatan relaksasi penuh yang asli.

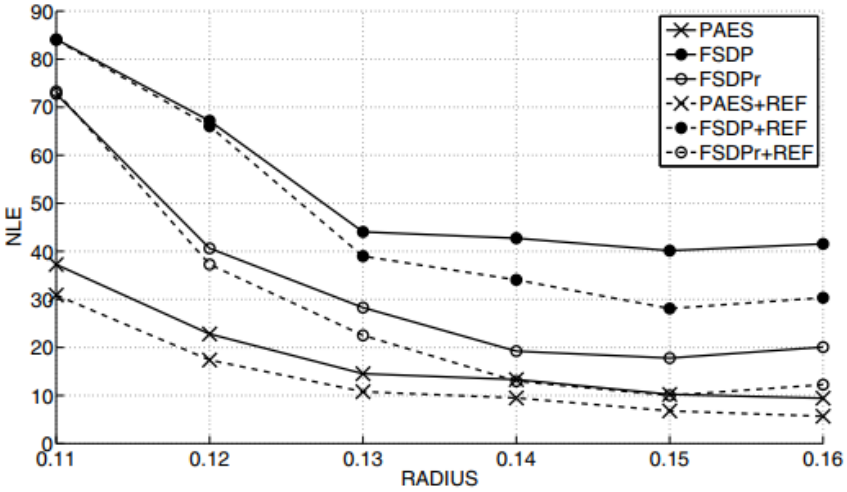
FSDPr menambahkan istilah regularisasi ke fungsi tujuan untuk mengurangi kecenderungan solusi SDP memiliki titik yang padat, yang terjadi setelah langkah terakhir memproyeksikan solusi SDP peringkat tinggi kembali ke bidang dua dimensi. Masalah utama dalam FSDPR adalah pilihan istilah regularisasi (λ). Untuk tujuan ini, kami telah mengadopsi strategi heuristik berikut. Diberikan sebuah skenario, pertama-tama kita memecahkan masalah non-regularisasi dan kemudian mengeksploitasi solusi non-regularisasi untuk menghitung batas atas λ (λ∗). λ∗ digunakan sebagai nilai awal dalam loop penyetelan utama, di mana masalah yang diatur diselesaikan; jika solusi terregularisasi saat ini tidak layak, maka nilai λ saat ini dibagi dua dan masalah terregularisasi baru diselesaikan lagi, hingga solusi layak diperoleh atau jumlah percobaan maksimum (MAX TRIES) tercapai. Dalam kasus terakhir, solusi teregulasi bertepatan dengan yang tidak teregulasi (yaitu λ = 0). Dalam percobaan kami, kami telah memperbaiki MAX TRIES = 5.

Terakhir, tujuan penyempurnaan berbasis gradien adalah untuk meningkatkan estimasi akhir yang diberikan oleh algoritme lokalisasi. Karena metode berbasis gradien umumnya tidak memberikan solusi optimal global ketika masalahnya tidak cembung, teknik ini dapat diterapkan sebagai fase fine-tuning setelah pendekatan solusi global telah. Dengan demikian, teknik ini dapat diterapkan pada metode pelokalan apa pun.

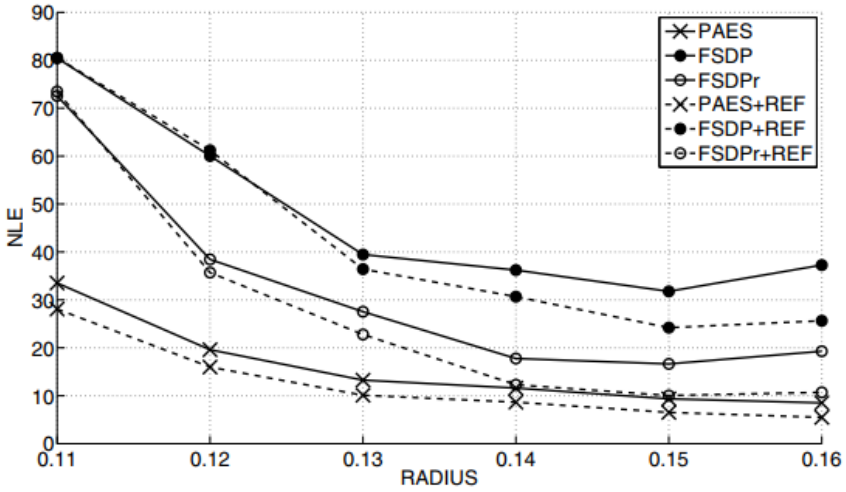
Dalam Gambar. 8.8–8.10, kami telah memplot sebagai garis padat NLE rata-rata yang diperoleh oleh algoritma FSDP, FSDPR dan PAES versus enam nilai radius R yang digunakan dalam percobaan. Selanjutnya, kami telah menunjukkan sebagai garis putus-putus NLE rata-rata diperoleh dengan menerapkan perbaikan gradien ke solusi akhir yang dihitung oleh tiga algoritma.



Gambar 8.8. Perbandingan antara PAES, FSDP dan FSDPr tanpa (garis padat) dan dengan (garis putus-putus) perbaikan gradien (REF) menggunakan 8% node sebagai node jangkar



Gambar 8.9 Perbandingan antara PAES, FSDP dan FSDPr tanpa (garis padat) dan dengan (garis putus-putus) perbaikan gradien (REF) menggunakan 10% node sebagai node jangkar



Gambar 8.10. Perbandingan antara PAES, FSDP dan FSDPr tanpa (garis padat) dan dengan (garis putus-putus) perbaikan gradien (REF) menggunakan 12% node sebagai node jangkar

Analisis angka-angka menyoroti bahwa, seperti yang diharapkan, FSDPr sedikit mengungguli FSDP. Selanjutnya, kami mengamati bahwa perbaikan gradien mampu meningkatkan estimasi hanya bila sudah cukup akurat. Memang, jika kami mempertimbangkan solusi yang dihasilkan oleh FSDP dan FSDPr untuk jari-jari konektivitas terendah (R = 0,11 dan R = 0,12 untuk FSDP, dan R = 0,11 untuk FSDPr), kami menyadari bahwa metode gradien tidak dapat mengganggu estimasi dari minimum lokal yang dicapai. Sebaliknya, ketika solusi ditandai dengan NLE rendah, metode gradien mampu memperbaikinya. Secara khusus, kesenjangan yang hampir konstan antara garis padat dan garis putus-putus untuk PAES menunjukkan peningkatan stabil yang diperkenalkan oleh fase penyempurnaan.

NLE yang diperoleh oleh PAES sebanding dengan yang diperoleh oleh *FSDPr+REF* ketika jari-jari konektivitas cukup tinggi (R 0,14), sementara NLE sedikit lebih rendah ketika R < 0,14.

Selanjutnya, dalam semua percobaan, PAES+REF secara signifikan mengungguli FSDPr+REF. Misalnya, ketika persentase simpul jangkar adalah 8% dan radius konektivitas adalah 0,11, dari Gambar 8.8 kita dapat memperoleh bahwa PAES+REF memberikan estimasi yang rata-rata 36,57% lebih akurat daripada FSDPR+REF. Persentase ini meningkat menjadi 45,75% ketika radius konektivitas sama dengan 0,16. Kesimpulan serupa dapat disimpulkan dengan menganalisis Gambar. 8.9–8.10, yang menunjukkan hasil yang diperoleh dengan menggunakan 10% dan 12% node jangkar: PAES+REF menghasilkan solusi yang 57,84% dan 53,72% (61,79% dan 48,87%) lebih akurat bila persentase simpul jangkar sama hingga 10% (12%) dan radius konektivitas adalah 0,11 dan 0,16.

8.8 KESIMPULAN

Penghematan energi adalah masalah yang sangat kritis dalam jaringan sensor nirkabel karena node sensor biasanya ditenagai oleh baterai dengan kapasitas terbatas. Karena radio adalah penyebab utama konsumsi daya di node sensor, pengiriman/penerimaan data dapat dibatasi melalui kompresi data. Dalam kerangka kerja ini, kami telah mengusulkan algoritma kompresi lossy yang sengaja dirancang untuk sumber daya terbatas yang tersedia pada node sensor papan dan berdasarkan skema modulasi kode pulsa diferensial di mana perbedaan antara sampel berurutan dikuantisasi. Karena proses kuantisasi mempengaruhi rasio kompresi dan kehilangan informasi, kami menerapkan NSGA-II untuk menghasilkan kombinasi yang berbeda dari parameter proses kuantisasi yang sesuai dengan trade-off optimal yang berbeda antara kinerja kompresi dan kehilangan informasi. Oleh karena itu, pengguna dapat memilih kombinasi dengan trade-off yang paling cocok untuk aplikasi tertentu. Kami menguji pendekatan kompresi lossy kami pada tiga kumpulan data yang dikumpulkan oleh WSN asli. Kami telah memperoleh rasio kompresi hingga 93,48% dengan kesalahan rekonstruksi yang sangat rendah. Selain itu, kami telah menunjukkan bagaimana pendekatan kami mengungguli LTC, algoritma kompresi lossy yang sengaja dirancang untuk disematkan di node sensor, baik dari segi rasio kompresi maupun kompleksitas.

Setelah informasi terkompresi mencapai sink, penting untuk mengetahui lokasi node di mana pengukuran tunggal telah dikumpulkan. Untuk mengaktifkan kesadaran lokasi kami telah mengusulkan untuk mengatasi masalah dengan mengeksploitasi algoritma evolusioner dua tujuan dan mengandalkan eksploitasi yang lebih baik dari grafik konektivitas sehingga dapat menentukan kendala topologi. Kendala seperti itu menentukan zona ruang di mana setiap sensor dapat atau tidak dapat ditemukan, sehingga mengurangi ruang pencarian dari algoritma evolusioner dan secara kontekstual kemungkinan membalik lokasi node secara ambigu. Selain itu kami telah membahas kemungkinan menggunakan teknik berbasis gradien standar yang mampu menyempurnakan estimasi akhir yang dihasilkan oleh algoritma evolusioner. Kami telah menunjukkan bahwa pendekatan yang diusulkan mampu memecahkan masalah lokalisasi dengan akurasi yang tinggi untuk sejumlah topologi yang berbeda, jari-jari konektivitas dan persentase node jangkar. Selanjutnya, kami telah membahas bagaimana pendekatan kami mengungguli teknik berbasis SDP standar dan versi regulernya.

DAFTAR PUSTAKA

Akram, A., Meredith, D., Allan, R.: Evaluation of BPEL to scientific workflows. In: Proc. of Sixth IEEE International Symposium on Cluster Computing and the Grid (CC- GRID 2006), pp. 269–274 (2006),

Akyildiz, I., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine 40(8), 102–114 (2002)

Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: a survey. Computer Networks 38, 393–422 (2002)

Albert, R., Barabasi, A.-L.: Statistical mechanics of complex networks. Reviews of Mod- ern Physics 74, 47–97 (2002)

Ali, S., Siegel, H., Maheswaran, M., Ali, S., Hensgen, D.: Task execution time model- ing for heterogeneous computing systems. In: Proc. of the Workshop on Heterogeneous Computing, pp. 185–199 (2000)

Alley, R., Soto, M., Kwark, L., Crocco, P., Koester, D.: Modeling and validation of on-die cooling of dual-core cpu using embedded thermoelectric devices. In: Twenty-fourth An- nual IEEE Semiconductor Thermal Measurement and Management Symposium, Semi- Therm 2008, pp. 77–82 (March 2008)

Aspnes, J., Goldenberg, D., Yang, Y.R.: On the Computational Complexity of Sensor Network Localization. In: Nikoletseas, S.E., Rolim, J.D.P. (eds.) ALGOSENSORS 2004. LNCS, vol. 3121, pp. 32–44. Springer, Heidelberg (2004)

Atienza, D., Valle, P.D., Paci, G., Poletti, F., Benini, L., Micheli, G.D., Mendias, J.M.: A Fast HW/SW FPGA-Based Thermal Emulation Framework for Multi-Processor System- on-Chip. In: Design Automation Conference (DAC), pp. 618–623 (2006)

Back, T., Fogel, D.B., Michalewicz, Z. (eds.): Evolutionary Computation, Part 1 and 2. IOP Publishing Ltd (2000)

Baker, R.J.: CMOS: circuit design, layout, and simulation, 2nd edn. Wiley (2008)

Bari, A., Wazed, S., Jaekel, A., Bandyopadhyay, S.: A genetic algorithm based approach for energy efficient routing in two-tiered sensor networks. Ad Hoc Networks 7(4), 665– 676 (2009)

Barr, K.C., Asanovic´, K.: Energy-aware lossless data compression. ACM Trans. Comput. Syst. 24(3), 250–291 (2006)

Barrenetxea, G., Ingelrest, F., Schaefer, G., Vetterli, M., Couach, O., Parlange, M.: SensorScope: Out-of-the-Box Environmental Monitoring. In: Proc. of the 7th Int. Conf. on Information Processing in Sensor Networks, pp. 332–343 (2008)

Bartschi Wall, M.:: A Genetic Algorithm for Resource-Constrained Scheduling. PhD Thesis, Massachusetts Institute of Technology, MA (1996)

Bash, C., Forman, G.: Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. HP Laboratories Palo Alto, Tech. Rep. HPL-2007-62 (August 2007)

Bashyal, S., Venayagamoorthy, G.: Collaborative routing algorithm for wireless sensor network longevity. In: 3rd IEEE International Conference on Intelligent Sensors, Sensor Networks and Information, ISSNIP 2007, pp. 515–520 (2007)

Beloglazov, A., Buyya, R., Lee, Y., Zomaya, A.Y.: A taxonomy and survey of energy- efficient data centers and cloud computing systems. Advances in Computers 82, 47–111 (2011)

Benini, L., Micheli, G.D.: Networks on Chips: A New SoC Paradigm. IEEE Com-puter 35(1), 70–78 (2002)

Bertozzi, D., Benini, L., De Micheli, G.: Error control schemes for on-chip communi-cation links: The energy-reliability trade off. IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. 24(6), 818–831 (2005)

Bevilacqua, A.: A Dynamic Load Balancing Metthod on a Heterogeneous Cluster of Workstations. Informatica 23(1), 49–56 (1999)

Bianchini, R., Carrera, E.V.: Analytical and Experimental Evaluation of Cluster-Based Network Servers. World Wide Web 3(4), 215–229 (2000)

Bianchini, R., Rajamony, R.: Power and Energy Management for Server Systems. Com- puter 37(11), 68–76 (2004)

Biswas, P., Lian, T.C., Wang, T.C., Ye, Y.: Semidefinite programming based algorithms for sensor network localization. ACM Trans. Sen. Netw. 2, 188–220 (2006)

Biswas, P., Liang, T.C., Toh, K.C., Ye, Y., Wang, T.C.: Semidefinite programming ap- proaches for sensor network localization with noisy distance measurements. IEEE Trans. Autom. Sci. Eng. 3(4), 360–371 (2006)

Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network local- ization. In: Proc. of the 3rd Int. Conf. on Information Processing in Sensor Networks, pp. 46–54 (2004)

Bleuler, S., Laumanns, M., Thiele, L., Zitzler, E.: PISA – A Platform and Programming Language Independent Interface for Search Algorithms. In: Fonseca, C.M., Fleming, P.J., Zitzler, E.,

Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 494–508. Springer, Heidelberg (2003)

Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm intelligence: from natural to artificial systems, vol. (1). Oxford University Press, USA (1999)

Borkar, S.: Design challenges of technology scaling. IEEE Micro 19(4), 23–29 (1999)

Brooks, D., Tiwari, V., Martonosi, M.: Wattch: a framework for architectural-level power analysis and optimizations. In: ISCA, pp. 83–94 (2000)

Brucker, P.: Scheduling Algorithms. Springer (2007)

Buchanan, M.: Nexus: Small Worlds and the Groundbreaking Theory of Networks. Nor- ton, W. W. & Company, Inc. (2003)

Buchbinder, N., Jain, N., Menache, I.: Online Job-Migration for Reducing the Electricity Bill in the Cloud. In: Domingo-Pascual, J., Manzoni, P., Palazzo, S., Pont, A., Scoglio, C. (eds.) NETWORKING 2011, Part I. LNCS, vol. 6640, pp. 172–185. Springer, Heidelberg (2011)

Burke, P.J., et al.: Quantitative Theory of Nanowire and Nanotube Antenna Performance. IEEE Transactions on Nanotechnology 5(4), 314–334 (2006)

Buyya, R., Murshed, M.: Gridsim: A toolkit for the modeling and simulation of dis- tributed resource management and scheduling for grid computing. Concurrency and Computation: Practice and Experience 14(13-15), 1175–1220 (2002)

Cabusao, G., Mochizuki, M., Mashiko, K., Kobayashi, T., Singh, R., Nguyen, T., Wu, P.: Data center energy conservation utilizing a heat pipe based ice storage system. In: 2010 IEEE CPMT Symposium Japan, pp. 1–4 (2010), doi:10.1109/CPMTSYMPJ.2010.5680287

Camilo, T., Carreto, C., Silva, J., Boavida, F.: An energy-efficient ant-based routing al- gorithm for wireless sensor networks. Ant Colony Optimization and Swarm Intelligence, 49–59 (2006)

Carrera, E.V., Pinheiro, E., Bianchini, R.: Conserving Disk Energy in Network Servers. In: The 17th Annual International Conference on Supercomputing (ICS 2003), pp. 86–97 (2003)

Carretero, J., Xhafa, F.: Using Genetic Algorithms for Scheduling Jobs in Large Scale Grid Applications. Journal of Technological and Economic Development –A Research Journal of Vilnius Gediminas Technical University 12(1), 11–17 (2006)

Chang, J., Tassiulas, L.: Maximum lifetime routing in wireless sensor networks. IEEE/ACM Transactions on Networking 12(4), 609–619 (2004)

Chang, M.F., et al.: CMP Network-on-Chip Overlaid With Multi-Band RF-Interconnect. In: Proc. of IEEE International Symposium on High-Performance Computer Architec- ture (HPCA), February 16-20, pp. 191–202 (2008)

Chang, M.F., et al.: RF Interconnects for Communications On-Chip. In: Proc. of Inter-national Symposium on Physical Design, April 13-16, pp. 78–83 (2008)

Chang, P.C., Wu, I.W., Shann, J.J., Chung, C.P.: ETAHM: An energy-aware task alloca- tion algorithm for heterogeneous multiprocessor. In: 45th ACM/IEEE Design Automa- tion Conference, pp. 776–779 (2008)

Chappel, D.A.: Enterprise Service Bus: Theory in Practice, 1st edn. (Hendrickson M). O'Reilly Media (2004)

Chen, G., Malkowski, K., Kandemir, M., Raghavan, P.: Reducing power with perfor- mance constraints for parallel sparse applications. In: 19th IEEE International on Parallel and Distributed Processing Symposium, 2005. Proceedings, p. 8 (2005)

Chen, W., Zhang, J.: An ant colony optimization approach to a grid workflow schedul- ing problem with various qos requirements. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 39(1), 29–43 (2009)

Cioara, T., Anghel, I., Salomie, I.: A Policy-based context aware self-management model. In: Proc. of the 11th International Symposium on Symbolic and Numeric Al- gorithms for Scientific Computing, Romania, (2009)

Circuits Multi-Projects, http://www.cmp.imag.fr

Cochran, R., Nowroz, A.N., Reda, S.: Post-silicon power characterization using thermal infrared emissions. In: ISLPED, pp. 331–336 (2010)

Cochran, R., Reda, S.: Consistent runtime thermal prediction and control through work- load phase detection. In: Design Automation Conference, DAC (2010)

Cochran, R., Reda, S.: Spectral techniques for high-resolution thermal characterization with limited sensor data. In: DAC, pp. 478–483 (2009)

Cohoon, J., Hegde, S., Martin, W., Richards, D.: Punctuated equilibria: a parallel genetic algorithm. In: Proceedings of the Second International Conference on Genetic Algo- rithms on Genetic Algorithms and their Application, pp. 148–154. L. Erlbaum Associates Inc. (1987)

Colajanni, M., Cardellini, V., Yu, P.S.: Dynamic Load Balancing in Geographically Dis- tributed Heterogeneous Web Servers. In: The 18th IEEE International Conference on Distributed Computing Systems (ICDCS 1998), pp. 295–303 (1998)

Coskun, A., Rosing, T., Gross, K.: Temperature management in multiprocessor socs us- ing online learning. In: 45th ACM/IEEE on Design Automation Conference, DAC 2008, pp. 890–893 (June 2008)

Coskun, A., Rosing, T., Gross, K.: Utilizing predictors for efficient thermal management in multiprocessor socs. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 28(10), 1503–1516 (2009)

Coskun, A.K., Atienza, D., Rosing, T.S., Brunschwiler, T., Michel, B.: Energy-efficient variable-flow liquid cooling in 3d stacked architectures. In: DATE, pp. 111–116 (2010)

Coskun, A.K., Rosing, T., Gross, K.: Proactive Temperature Balancing for Low-Cost Thermal Management in MPSoCs. In: International Conference on Computer-Aided De- sign (ICCAD), pp. 250–257 (2008)

Coskun, A.K., Rosing, T.V., Gross, K.C.: Utilizing Predictors for Efficient Thermal Man- agement in Multiprocessor SoCs. IEEE Transactions on CAD 28, 1503–1516 (2009)

Costa, J.A., Patwari, N., Hero III, A.O.: Distributed weighted–multidimensional scaling for node localization in sensor networks. ACM Trans. on Sensor Networks 2(1), 39–64 (2006)

Croce, S., Marcelloni, F., Vecchio, M.: Reducing power consumption in wireless sensor networks using a novel approach to data aggregation. The Computer Journal 51(2), 227–239 (2008)

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6(2), 182–197 (2002)

Deelman, E., Blythe, J., Gil, Y., Kesselman, C., Mehta, G., Patil, S., Su, M.-H., Vahi, K., Livny, M.: Pegasus: Mapping Scientific Workflows onto the Grid. In: Dikaiakos, M.D. (ed.) AxGrids 2004. LNCS, vol. 3165, pp. 11–20. Springer, Heidelberg (2004)

Donoho, D.L., Johnstone, I.M.: Ideal spatial adaptation by wavelet shrinkage. Biometrika 81(3), 425–455 (1994)

Duato, J., et al.: Interconnection Networks - An Engineering Approach. Morgan Kauf- mann (2002)

Durresi, A., Durresi, M., Paruchuri, V., Barolli, L.: Ad Hoc Communications for Emer-gency Conditions. In: The 25th IEEE International Conference on Advanced Information Networking and Applications (AINA 2011), pp. 787–794 (2011)

Dustdar, S., Dorn, C., Li, F., Baresi, L., Cabri, G., Pautasso, C., Zambonelli, F.: A roadmap towards sustainable self-aware service systems. In: Proc. of the 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2010), pp. 10–19. ACM (2010)

Enokido, T., Aikebaier, A., Misbah Deen, S., Takizawa, M.: Power Consumption-based Server Selection Algorithms for Communication-based Systems. In: The 13th Interna- tional Conference on Network-based Information Systems (NBiS 2010), pp. 201–208 (2010)

Enokido, T., Aikebaier, A., Takizawa, M.: A Model for Reducing Power Consumption in Peer-to-Peer Systems. IEEE Systems Journal 4(2), 221–229 (2010)

Enokido, T., Aikebaier, A., Takizawa, M.: Process Allocation Algorithms for Sav- ing Power consumption in Peer-to-Peer Systems. IEEE Trans. on Industrial Electron- ics 58(6), 2097–2105 (2011)

Enokido, T., Suzuki, K., Aikebaier, A., Takizawa, M.: Algorithms for Reducing the Total Power Consumption in Data Communication-based Applications. In: The 24th IEEE International Conference on Advanced Information Networking and Applications (AINA 2010), pp. 142–149 (2010)

Enokido, T., Takizawa, M.: A Purpose-based Synchronization Protocol for Secure In- formation Flow Control. Journal of Computer Systems Science and Engineering (JC- SSE) 25(2), 25–32 (2010)

Eriksson, R., Olsson, B.: On the performance of evolutionary algorithms with life-time adaptation in dynamic fitness landscapes, Congress. Evolutionary Computation 2, 1293–1300 (2004)

Fan, X., Weber, W., Barroso, L.A.: Power Provisioning for a Warehouse-sized Computer. In: The 34th International Symposium on Computer Architecture (ICSA 2007), pp. 13– 23 (2007)

Ferraiolo, D.F., Kuhn, D., Chandramouli, R.: Role-Based Access Control. Artech Hous, Norwood (2007)

Ferrie`re, H.D., Fabre, L., Meier, R., Metrailler, P.: TinyNode: a comprehensive platform for wireless sensor network applications. In: IPSN 2006: Proc. of the 5th Int. Conf. on Information Processing in Sensor Networks, pp. 358–365 (2006)

Fibich, P., Matyska, L., Rudova´, H.: Model of grid scheduling problem. In: Proc. of the AAAI 2005 Workshop on Exploring Planning and Scheduling for Web Services, Grid and Autonomic Computing (2005)

Fischer, M.J., Lynch, N.A., Paterson, M.S.: Impossibility of Distributed Consensus with One Faulty Process. Journal of the ACM 32(2), 374–382 (1985)

Floyd, B.A., et al.: Intra-Chip Wireless Interconnect for Clock Distribution Implemented With Integrated Antennas, Receivers, and Transmitters. IEEE Journal of Solid-State Cir- cuits 37(5), 543–552 (2002)

Freeh, V., Lowenthal, D.: Using multiple energy gears in mpi programs on a power- scalable cluster. In: Proceedings of the tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, pp. 164–173 (2005)

Freitag, M., et al.: Hot carrier electroluminescence from a single carbon nanotube. Nano Letters 4(6), 1063–1066 (2004)

Friedrich, G., Fugini, M., Mussi, E., Pernici, B., Tagni, G.: Exception han-dling for repair in service-based processes. IEEE Transactions on Soft-ware Engineering 36(2), 198–215 (2010)

Fu, B., Ampadu, P.: Error control combining Hamming and product codes for energy efficient nanoscale on-chip interconnects. IET Computers & Digital Techniques 4(3), 251–261 (2010)

Fukuda, M., et al.: A 0.18 m CMOS Impulse Radio Based UWB Transmitter for Global Wireless Interconnections of 3D Stacked-Chip System. In: Proc. of International Con- ference Solid State Devices and Materials, pp. 72–73 (September 2006)

GAMES project website, http://www.green-datacenters.eu

Ganguly, A., et al.: A Unified Error Control Coding Scheme to Enhance the Reliability of a Hybrid Wireless Network-on-Chip. In: Proc. of the IEEE Defect and Fault Tolerance Symposium (DFTS), Vancouver, Canada (October 2011)

Ganguly, A., et al.: Crosstalk-Aware Channel Coding Schemes for Energy Efficient andReliable NoC Interconnects. IEEE Transactions on VLSI (TVLSI) 17(11), 1626–1639 (2009)

Ganguly, A., et al.: Scalable Hybrid Wireless Network-on-Chip Architectures for Multi-Core Systems. IEEE Transactions on Computers (TC) 60(10), 1485–1502 (2011)

Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-completeness. WH Freeman & Co. (1979)

Garg, S., Buyya, R., Segel, H.: Scheduling parallel aplications on utility grids: Time and cost trade-off management. In: Mans, B. (ed.) Proc. of the 32nd ACSC, Wellington, Australia. CRPIT, vol. 91 (2009)

Ge, Y., Malani, P., Qiu, Q.: Distributed task migration for thermal management in many- core systems. In: 2010 47th ACM/IEEE on Design Automation Conference (DAC), pp. 579–584 (June 2010)

Ge, Y., Qiu, Q.: Dynamic thermal management for multimedia applications using ma- chine learning. In: 2011 48th ACM/EDAC/IEEE on Design Automation Conference (DAC), pp. 95–100 (June 2011)

Ghemawat, S., Gobioff, H., Leung, S.: The Google File System. In: The 19th ACM Symposium on Operating Systems Principles (SOSP 2003), pp. 29–43 (2003)

Go¨rlach, K., Sonntag, M., Karastoyanova, D., Leymann, F., Reiter, M.: Conventional simulation workflow technology for scientific simulation. In: Yang, X., Wang, L., Jie, W. (eds.) Guide to e-Science, pp. 0–31 (2011)

Goldberg, D.E., Lingle Jr., R.: Alleles, loci and the travelling salesman problem. In: Proc. of the ICA 1985, Pittsburgh (1985)

Gosain, S., Malhotra, A., El Sawy, O.A.: Coordinating for flexibility in e-business supply chains. Journal of Management Information Systems 21(3), 7–45 (2004)

Gosh, P., Das, S.: Mobility-aware cost-efficient job scheduling for single-class grid jobs in a generic mobile grid architecture. Future Generation Computer Systems 26(8), 1356–1367 (2010)

Graham, R., Lawler, E., Lenstra, J., Rinnooy Kan, A.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Annals of Discrete Mathematics 5, 287–326 (1979)

Green, W.M.J., et al.: Ultra-compact, low RF power, 10Gb/s silicon Mach-Zehnder mod- ulator. Optics Express 15(25), 17106–17113

Gronowski, P., Bowhill, W., Preston, R., Gowan, M., Allmon, R.: High-performance microprocessor design. IEEE Journal of Solid-State Circuits 33(5), 676–686 (1998)

Grossman, R.L.: The Case for Cloud Computing. IT Professional 11(2), 23–27 (2009)

Grueninger, T.: Multimodal optimization using genetic algorithms. Technical report, Department of Mechanical Engineering. MIT, Cambridge (1997)

Gruian, F., Kuchcinski, K.: Lenes: task scheduling for low-energy systems using variable supply voltage processors. In: Proceedings of the 2001 Asia and South Pacific Design Automation Conference, pp. 449–455 (2001)

Gupta, G., Younis, M.: Load-balanced clustering of wireless sensor networks. In: IEEE International Conference on Communications, ICC 2003, vol. 3, pp. 1848–1852 (2003)

Gupta, S.K.S., Mukherjee, T., Varsamopoulos, G., Banerjee, A.: Research directions in energy-sustainable cyber-physical systems. Elsevier Comnets Special Issue in Sustain- able Computing (SUSCOM), Invited Paper 1(1), 57–74 (2011)

Guzek, K., Pecero, J.E., Dorrosoro, B., Bouvry, P., Khan, S.U.: A cellular genetic al- gorithm for scheduling applications and energy-aware communication optimization. In: Proc. of the ACM/IEEE/IFIP Int. Conf. on High Performance Computing and Simulation (HPCS), Caen, France, pp. 241–248 (2010)

Hamann, H.F., Lo´pez, V., Stepanchuk, A.: Thermal zones for more efficient data center energy management. In: 12th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), pp. 1–6 (2010), doi:10.1109/ITHERM.2010.5501332

Hang, Y., Kuo, J., Ahmad, I.: Energy efficiency in data centers and cloud-based multimedia services: An overview and future directions. In: 2010 International Green Computing Conference, pp. 375–382 (2010), doi:10.1109/GREENCOMP.2010.5598292

Hanson, G.W.: On the Applicability of the Surface Imped-ance Integral Equation for Optical and Near Infrared Copper Dipole Antennas. IEEE Transactions on Antennas and Propagation 54(12), 3677–3685 (2006)

Harvill, L.M.: Standard Error of Measurement. East Tennesse State University, http:// www.ncme.org/pubs/items/16.pdf

Heath, T., Diniz, B., Carrera, E.V., Meira, W.J., Bianchini, R.: Energy Conservation in Heterogeneous Server Clusters. In: The 10th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP 2005), pp. 186–195 (2005)

Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, p. 10 (2002)

Hemmert, S.: Green HPC: From Nice to Necessity. Computing in Science and Engineer- ing 12(6), 8–10 (2010)

Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. The MIT Press (1992)

Horn, J., Nafpliotis, N., Goldberg, D.E.: A Niched Pareto Genetic Algorithm for Multi- objective Optimization. In: Proc. of the 1st IEEE World Congress on Evolutionary Com- putation, vol. 1, pp. 82–87 (1994)

Hsiao, M.Y.: A class of optimal minimum odd-weight-column SEC-DED codes. IBM J. Res. Dev. 14(4), 395–401 (1970)

Huang, R., Chen, Z., Xu, G.: Energy-aware routing algorithm in wsn using predication- mode. In: 2010 IEEE International Conference on Communications, Circuits and Sys- tems (ICCCAS), pp. 103–107 (2010)

Huang, W., Stan, M.R., Skadron, K., Sankaranarayanan, K., Ghosh, S., Velusam, S.: Compact thermal modeling for temperature-aware design. In: Proceedings of the 41st Annual Design Automation Conference, DAC 2004, pp. 878–883. ACM, New York (2004)

Huang, Y., et al.: Performance Prediction of Carbon Nano-tube Bundle Dipole Antennas. IEEE Transactions on Nanotechnology 7(3), 331–337 (2008)

Huffman, D.: A method for the construction of minimum-redundancy codes. In: Proc. of the IRE, vol. 40(9), pp. 1098–1101 (1952)

Humphrey, M., Thompson, M.: Security implications of typical grid computing usage scenarios. In: Proc. of the Conf. on High Performance Distributed Computing (2001)

Hung, W.-L., Link, G.M., Xie, Y., Vijaykrishnan, N., Irwin, M.J.: Interconnect and thermal-aware floorplanning for 3d microprocessors. In: ISQED, pp. 98–104 (2006)

Hwang, S., Kesselman, C.: A flexible framework for fault tolerance in the grid. J. of Grid Computing 1(3), 251–272 (2003)

Ikeda, M., Kulla, E., Hiyama, M., Barolli, L., Takizawa, M.: Experimental Results of a MANET Testbet in Indoor Stairs Environment. In: The 25th IEEE International Confer- ence on Advanced Information Networking and Applications (AINA 2011), pp. 779–786 (2011)

Inoue, T., Ikeda, M., Enokido, T., Aikebaier, A., Takizawa, M.: A Power Consumption Model for Storage-based Applications. In: The Fifth International Conference on Com- plex, Intelligent, and Software Intensive Systems, CISIS 2011 (2011)

Intel Corporation.: Intel Xeon Processor 5600 Series : The Next Generation of Intelligent Server Processors. Available via DIALOG, http://www.intel.com/content/www/us/en/processors/xeon/xeon-5600-brief.html (cited November 8, 2011)

Isci, C., Martonosi, M.: Runtime power monitoring in high-end processors: Methodol- ogy and empirical data. In: Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 36, p. 93 (2003)

Islam, O., Hussain, S.: An intelligent multi-hop routing for wireless sensor networks. In: 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops, WI-IAT 2006 Workshops, pp. 239–242 (2006)

Ismail, A., Abidi, A.: A 3 to 10GHz LNA Using a Wide-band LC-ladder Matching Net- work. In: Proc. of IEEE International Solid-State Circuits Conference, February 15-19, pp. 384–534 (2004)

Iyengar, M., Schmidt, R., Caricari, J.: Reducing energy usage in data centers through control of Room Air Conditioning units. In: 12th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), pp. 1–11 (2010), doi:10.1109/ITHERM.2010.5501418

Ji, X., Zha, H.: Sensor positioning in wireless ad-hoc sensor networks using multidimen- sional scaling. In: INFOCOM 2004: Proc. of the 23rd Annual Joint Conf. of the IEEE Computer and Communications Societies, vol. 4, pp. 2652–2661 (2004)

Job Scheduling Algorithms in Linux Virtual Server. Available via DIALOG, http:// www.linuxvirtualserver.org/docs/scheduling.html (cited August 29, 2011)

Kannan, A.A., Fidan, B., Mao, G.: Analysis of flip ambiguities for robust sensor network localization. IEEE Trans. Veh. Technol. 59(4), 2057–2070 (2010)

Kannan, A.A., Mao, G., Vucetic, B.: Simulated annealing based wireless sensor network localization with flip ambiguity mitigation. In: VTC 2006: Proc. of the 63rd IEEE Ve- hicular Technology Conf., pp. 1022–1026 (2006)

Kansal, A., Hsu, J., Srivastava, M., Raghunathan, V.: Harvesting aware power manage- ment for sensor networks. In: Proceedings of the 43rd annual Design Automation Con- ference, pp. 651–656. ACM (2006)

Kant, K.: Data center evolution: A tutorial on state of the art, issues, and challenges. Computer Networks 53(17), 2939–2965 (2009)

Karaenke, P., Schuele, M., Micsik, A., Kipp, A.: Inter-organizational Interoperability through Integration of Multiagent, Web Service, and Semantic Web Technologies. In: Fischer, K., Mu¨ller, J.P., Levy, R. (eds.) ATOP 2009 and ATOP 2010. LNBIP, vol. 98, pp. 55–75. Springer, Heidelberg (2012)

Kempa, K., et al.: Carbon Nanotubes as Optical Antennae. Advanced Materials 19, 421– 426 (2007)

Kessaci, Y., Mezmaz, M., Melab, N., Talbi, E.-G., Tuyttens, D.: Parallel Evolutionary Algorithms for Energy Aware Scheduling. In: Bouvry, P., Gonza´lez-Ve´lez, H., Kołodziej, J. (eds.) Intelligent Decision Systems in Large-Scale Distributed Environments. SCI, vol. 362, pp. 75–100. Springer, Heidelberg (2011)

Khan, M.A., Hankendi, C., Coskun, A.K., Herbordt, M.C.: Software optimization for performance, energy, and thermal distribution: Initial case studies. In: IEEE Workshop on Thermal Modeling and Management: From Chips to Data Centers (in conj. with Green Computing Conference (IGCC)) (2011)

Khan, S.: A goal programming approach for the joint optimization of energy con- sumption and response time in computational grids. In: Proc. of the 28th IEEE Inter- national Performance Computing and Communications Conference (IPCCC), Phoenix, AZ, USA, pp. 410–417 (2009a)

Khan, S.: A self-adaptive weighted sum technique for the joint optimization of perfor- mance and power consumption in data centers. In: Proc. of the 22nd International Con- ference

on Parallel and Distributed Computing and Communication Systems (PDCCS), USA, pp. 13–18 (2009b)

Khan, S.U., Ahmad, I.: A cooperative game theoretical technique for joint optimization of energy consumption and response time in computational grids. IEEE Tran. on Parallel and Distributed Systems 20(3), 346–360 (2009)

Kim, S., Kojima, M., Waki, H.: Exploiting sparsity in SDP relaxation for sensor network localization. SIAM J. Optim. 20, 192–215 (2009)

Kimura, H., Sato, M., Hotta, Y., Boku, T., Takahashi, D.: Emprical study on reducing energy of parallel programs using slack reclamation by dvfs in a power-scalable high performance cluster. In: 2006 IEEE International Conference on Cluster Computing, pp. 1–10 (2006)

Kipp, A., Liu, J., Jiang, T.: Testbed architecture for generic, energy-aware evaluations and optimisations. In: Proc. of the First International Conference on Advanced Communications and Computation (INFOCOMP 2011), Barcelona (2011) (article in press)

Klusacˇek, D., Rudova´, H.: Efficient grid scheduling through the incremental schedule- based approach. Computational Intelligence 27(1), 4–22 (2011)

Knowles, J.D., Corne, D.W.: Approximating the nondominated front using the Pareto Archived Evolution Strategy. IEEE Trans. Evol. Comput. 8(2), 149–172 (2000)

Koduru, P., Dong, Z., Das, S., Welch, S., Roe, J., Charbit, E.: A Multi objective Evolutionary-Simplex Hybrid Approach for the Optimization of Differential Equation Models of Gene Networks. IEEE Trans. Evolutionary Computation 12(5), 572–590 (2008)

Kołodziej, J., Xhafa, F.: Enhancing the genetic-based scheduling in computational grids by a structured hierarchical population. Future Generation Computer Systems 27, 1035–1046 (2011)

Kołodziej, J., Xhafa, F.: Integration of task abortion and security requirements in ga- based meta-heuristics for independent batch grid scheduling. Computers and Mathemat- ics with Applications 63, 350–364 (2011)

Kołodziej, J., Xhafa, F.: Meeting security and user behaviour requirements in grid scheduling. Simulation Modelling Practice and Theory 19(1), 213–226 (2011)

Kołodziej, J., Xhafa, F.: Modern approaches to modelling user requirements on resource and task allocation in hierarchical computational grids. Int. J. on Appled Mathematics and Computer Science 21(2), 243–257 (2011)

Kołodziej, J.: Evolutionary Hierarchical Multi-Criteria Metaheuristics for Scheduling in Large-Scale Grid Systems. SCI, vol. 419. Springer, Heidelberg (2012)

Kulkarni, V., Forster, A., Venayagamoorthy, G.: Computational intelligence in wire- less sensor networks: A survey. IEEE Communications Surveys & Tutorials (99), 1–29 (2011)

Kumar, A., et al.: Toward Ideal On-Chip Communication Using Express Virtual Chan-nels. IEEE Micro 28(1), 80–90 (2008)

Kwasinski, A., Kudithipudi, D.: Towards integrated circuit thermal profiling for reduced power consumption: Evaluation of distributed sensing techniques. In: Proceedings of the International Conference on Green Computing, GREENCOMP 2010, pp. 503–508. IEEE Computer Society, Washington, DC (2010)

Kwok, Y.-K., Hwang, K., Song, S.: Selfish grids: Game-theoretic modeling and nas/psa benchmark evaluation. IEEE Tran. on Parallel and Distributing Systems 18(5), 1–16 (2007)

Lapin, L.: Probability and Statistics for Modern Engineering, 2nd edn. (1998)

Laszewski, G., Von Wang, L., Younge, A.J., He, X.: Power-aware scheduling of virtual machines in DVFS-enabled clusters. In: IEEE Intl. Conf. on Cluster Computing and Workshops, pp. 1–10 (2009), doi:10.1109/CLUSTR.2009.5289182

Le, K., Bilgir, O., Bianchini, R., Martonosi, M., Nguyen, T.: Managing the cost, energy consumption, and carbon footprint of Internet services. SIGMETRICS Perform. Eval. Rev. 38(1), 357–358 (2010)

Lee, B.G., et al.: Ultrahigh-Bandwidth Silicon Photonic Nan-owire Waveguides for On- Chip Networks. IEEE Photonics Technology Letters 20(6), 398–400 (2008)

Lee, Y., Zomaya, A.: A novel state transition method for metaheuristic-based schedul- ing in heterogeneous computing systems. IEEE Transactions on Parallel and Distributed Systems, 1215–1223 (2008)

Lee, Y., Zomaya, A.: Minimizing energy consumption for precedence-constrained ap- plications using dynamic voltage scaling. In: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, pp. 92–99 (2009)

Lee, Y.C., Zomaya, A.Y.: Minimizing energy consumption for precedence-constrained applications using dynamic voltage scaling. In: Proc. of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid CCGrid), Shanghai, China, pp. 92–99 (2009)

Li, G., Muthusamy, V., Jacobsen, H.: A distributed service-oriented architecture for busi- ness process execution. ACM Transactions on the Web 4(1), 1–33 (2010), http:// portal.acm.org/citation.cfm?doid=1658373.1658375

Li, K.: Performance Analysis of Power-Aware Task Scheduling Algorithms on Multi-processor Computers with Dynamic Voltage and Speed. IEEE Trans. on Parallel and Distributed Systems 19(11), 1484–1497 (2008), doi:10.1109/TPDS.2008.122

Li, S., Ahn, J.H., Strong, R.D., Brockman, J.B., Tullsen, D.M., Jouppi, N.P.: Mcpat: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In: MICRO 42: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture, pp. 469–480 (2009)

Liefooghe, A., Jourdan, L., Talbi, E.: A unified model for evolutionary multi-objective optimization and its implementation in a general purpose software framework. In: IEEE Symposium on Computational Intelligence in Miulti-Criteria Decision-Making, MCDM 2009, pp. 88–95. IEEE (2009)

Lim, M., Freeh, V., Lowenthal, D.: Adaptive, transparent frequency and voltage scaling of communication phases in mpi programs. In: Proceedings of the ACM/IEEE, 2006 Conference, SC 2006, pp. 14–14 (2006)

Lindsey, S., Raghavendra, C.: Pegasis: Power-efficient gathering in sensor information systems. In: IEEE Aerospace Conference Proceedings, vol. 3, pp. 3–1125 (2002)

Link, G.M., Vijaykrishnan, N.: Thermal trends in emerging technologies. In: Proceedings of the 7th International Symposium on Quality Electronic Design, ISQED 2006, pp. 625– 632. IEEE Computer Society, Washington, DC (2006)

Liu, X., Zhao, H., Li, X.: EPC: Energy-aware Probability-based Clustering Algorithm for Correlated Data Gathering in Wireless Sensor Networks. In: The 25th IEEE International Conference on Advanced Information Networking and Applications (AINA 2011), pp. 419–426 (2011)

Liu, Y., Yang, H., Luo, R., Wang, H.: Combining Genetic Algorithms Based Task Map-ping and Optimal Voltage Selection for Energy-Efficient Distributed System Synthesis. In: 2006 Intl. Conf. on Communications, Circuits and System, vol. 3, pp. 2074–2078 (2006), doi:10.1109/ICCCAS.2006.285087

Liu, Z., Lin, M., Wierman, A., Low, S.H., Andrew, L.L.H.: Greening geographical load balancing. In: Proc. ACM SIGMETRICS. ACM, San Jose (2011)

Lojek, B.: Reflectivity of the Silicon Semiconductor Substrate and its Dependence on the Doping Concentration and Intensity of the Irradiation. In: Proc. of the 11th IEEE International Conference on Advanced Thermal Processing of Semiconductors, pp. 215–220 (2003)

Lu, Z., et al.: Connection-oriented Multicasting in Wormhole-switched Networks on Chip. In: Proc. of IEEE Computer Society Annual Symposium on VLSI, pp. 205–210 (2006)

Luda¨scher, B., Altintas, I., Berkley, C.: Scientific workflow man-agement and the Ke- pler system. Concurrency and Computation: Practice and Experience 18(10), 1039–1065 (2006), doi.wiley.com/10.1002/cpe.994

Luda¨scher, B., Weske, M., McPhillips, T., Bowers, S.: Scientific Workflows: Business as Usual? In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 31–47. Springer, Heidelberg (2009)

Maniezzo, V., Carbonaro, A.: Ant colony optimization: an overview. In: Essays and Sur- veys in Metaheuristics, pp. 21–44 (2001)

Mao, G., Fidan, B., Anderson, B.D.O.: Wireless sensor network localization techniques. Computer Networks 51(10), 2529–2553 (2007)

Marcelloni, F., Vecchio, M.: A simple algorithm for data compression in wireless sensor networks. IEEE Commun. Lett. 12(6), 411–413 (2008)

Marcelloni, F., Vecchio, M.: A two-objective evolutionary approach to design lossy compression algorithms for tiny nodes of wireless sensor networks. Evolutionary Intelligence 3, 137–153 (2010)

Marcelloni, F., Vecchio, M.: An efficient lossless compression algorithm for tiny nodes of monitoring wireless sensor networks. The Computer Journal 52(8), 969–987 (2009)

Marcelloni, F., Vecchio, M.: Enabling energy-efficient and lossy-aware data compression in wireless sensor networks by multi-objective evolutionary optimization. Information Sciences 180(10), 1924–1941 (2010)

Marinis, T.S., et al.: Wafer level vacuum packaging of MEMS sensors. In: Proc. of Elec- tronic Components and Technology Conference, May 31-June 3, vol. 2, pp. 1081–1088 (2005)

Meisner, D., Gold, B.T., Wenisch, T.F.: PowerNap: Eliminating Server Idle Power. In: The 14th International Conference on Architectural Support for Programming Lan- guages and Operating Systems (ASPLOS 2009), pp. 205–216 (2009)

Mejia-Alvarez, P., Levner, E., Mosse´, D.: Adaptive scheduling server for power-aware real-time tasks. ACM Trans. Embed. Comput. Syst. 3(2), 284–306 (2004)

Mesa-Martinez, F.J., Nayfach-Battilana, J., Renau, J.: Power model validation through thermal measurements. In: ISCA, pp. 302–311 (2007)

Mezmaz, M., Melab, N., Kessaci, Y., Lee, Y., Talbi, E., Zomaya, A., Tuyttens, D.: A par- allel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. Journal of Parallel and Distributed Computing (2011)

Mezmaz, M., Melab, N., Kessaci, Y., Lee, Y., Talbi, E.-G., Zomaya, A., Tuyttens, D.: A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud com- puting systems. J. Parallel Distrib. Comput. (2011), doi:10.1016/j.jpdc.2011.04.007

Miao, L., Qi, Y., Hou, D., Dai, Y., Shi, Y.: A multi-objective hybrid genetic algorithm for energy saving task scheduling in cmp system. In: Proc. of IEEE Int. Conf. on Systems, Man and Cybernetics (ICSMC 2008), pp. 197–201 (2008)

Miao, L., Qi, Y., Hou, D., Dai, Y.: Energy-Aware Scheduling Tasks on Chip Multipro-cessor. In: Third International Conference on Natural Computation, vol. 4, pp. 319–323 (2007), doi:10.1109/ICNC.2007.356

Miao, L., Qi, Y., Hou, D., Dai, Y.H., Shi, Y.: "A multi-objective hybrid genetic algorithm for energy saving task scheduling in CMP system. In: IEEE Intl. Conf. on Systems, Man and Cybernetics, pp. 197–201 (2008), doi:10.1109/ICSMC.2008.4811274

Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer (1992)

Michalewicz, Z.: Genetic algorithms + data structures = evolution programs, 2nd edn. Springer, New York (1994)

Mitchell, M.: An introduction to genetic algorithms. The MIT press (1998)

Modafferi, S., Mussi, E., Maurino, A., Pernici, B.: A framework for provisioning of complex e-services. In: Proc. of the IEEE International Conference on Services Com- puting (SCC 2004), pp. 81–90. IEEE Press (2004), http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1357993

Moore, J., Chase, J., Ranganathan, P., Sharma, R.: Making scheduling "cool": temperature-aware workload placement in data centers. In: ATEC 2005: Proceedings of the Annual Conference on USENIX Annual Technical Conference, pp. 5–5. USENIX Association, Berkeley (2005)

Moore, J., Chase, J., Ranganathan, P.: Weatherman: Automated, online, and predictive thermal mapping and management for data centers. In: IEEE International Conference on Autonomic Computing (ICAC), pp. 155–164 (June 2006)

Muhammad, R., Hussain, F., Hussain, O.: Neural network-based approach for predicting trust values based on non-uniform input in mobile applications. The Computer Journal Online (2011)

Mukherjee, R., Memik, S.O.: Systematic temperature sensor allocation and placement for microprocessors. In: Proceedings of the 43rd Annual Design Automation Conference, DAC 2006, pp. 542–547. ACM, New York (2006)

Mukherjee, T., Banerjee, A., Varsamopoulos, G., Gupta, S.K.S., Rungta, S.: Spatio- temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers. Computer Networks (June 2009), http://dx.doi.org/10.1016/j.comnet.2009.06.008

Murali, S., De Micheli, G., Benini, L., Theocharides, T., Vijaykrishnan, N., Irwin, M.: Analysis of error recovery schemes for networks on chips. IEEE Design Test Com- put. 22(5), 434–442 (2005)

Murphy, R., Sterling, T., Dekate, C.: Advanced Architectures and Execution Models to Support Green Computing. Computing in Science and Engineering 12(6), 38–47 (2010)

Noman, N., Iba, H.: Accelerating Differential Evolution Using an Adaptive Local Search. IEEE Trans. Evolutionary Computation 12(1), 107–125 (2008)

O'Neal, J.: Differential pulse-code modulation (PCM) with entropy coding. IEEE Trans. Inf. Theory 22(2), 169–174 (1976)

Ogras, U.Y., Marculescu, R.: "It's a Small World After All": NoC Performance Opti- mization Via Long-Range Link Insertion. IEEE Transactions on Very Large Scale Inte- gration (VLSI) Systems 14(7), 693–706 (2006)

Oinn, T., Greenwood, M., Addis, M.: Taverna: lessons in creating a workflow environment for the life sciences. Concurrency and Computation: Practice and Experi- ence 18(10), 1067–1100 (2006), doi.wiley.com/10.1002/cpe.993

Olivier, I., Smith, D., Holland, J.: A study of permutation crossover operators on the travelling salesman problem. In: Proc. of the ICGA 1987, Cambridge, MA, pp. 224–230 (1987)

Page, A., Naughton, J.: Dynamic Task Scheduling using Genetic Algorithms for Hetero- geneous Distributed Computing. In: 19th IEEE Intl. Conf. on Parallel and Distributed Processing (2005), doi:10.1109/IPDPS.2005.184

Pakbaznia, E., Ghasemazar, M., Pedram, M.: Temperature-aware dynamic resource pro- visioning in a power-optimized datacenter. In: Design, Automation Test in Europe Con- ference Exhibition (DATE), pp. 124–129 (March 2010)

Pande, P.P., et al.: Performance Evaluation and Design Trade-offs for Network-on-chip Interconnect Architectures. IEEE Transactions on Computers 54(8), 1025–1040 (2005)

Patel, K.N., Markov, I.L.: Error-correction and crosstalk avoidance in DSM busses. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 12(10), 1076–1080 (2004)

Patwari, N., Ash, J.N., Kyperountas, S., Hero III, A.O., Moses, R.L., Correal, N.S.: Lo- cating the nodes: cooperative localization in wireless sensor networks. IEEE Signal Pro- cess. Mag. 22(4), 54–69 (2005)

Pavlidis, V.F., Friedman, E.G.: 3-D Topologies for Networks-on-Chip. IEEE Transac- tions on Very Large Scale Integration (VLSI) 5(10), 1081–1090 (2007)

Pedram, M., Nazarian, S.: Thermal modeling, analysis, and management in vlsi circuits: Principles and methods. Proceedings of the IEEE 94(8), 1487–1501 (2006)

Peralta, L.M.R.: Collaborative localization in wireless sensor networks. In: SENSOR- COMM 2007: Proc. of the 2007 Int. Conf. on Sensor Technologies and Applications, pp. 94–100 (2007)

Perera, G.: New Search Paradigms and Power Management for Peer-to-Peer File Shar- ing. VDM Verlag, Saarbrucken (2008)

Pinel, F., Pecero, J., Bouvry, P., Khan, S.U.: Memory-aware green scheduling on multi- core processors. In: Proc. of the 39th IEEE International Conference on Parallel Process- ing (ICPP), pp. 485–488 (2010)

Postel, J., Reynolds, J.: RFC: 959 File transfer protocol (FTP), Available via DIALOG, http://www.ietf.org/rfc/rfc959.txt (cited November 7, 2011)

Pradhan, S., Kusuma, J., Ramchandran, K.: Distributed compression in a dense mi- crosensor network. IEEE Signal Process. Mag. 19(2), 51–60 (2002)

Price, K., Storn, R., Lampinen, J.: Differential evolution: a practical approach to global optimization. Springer (2005)

Quick start guide to increase data center energy efficiency, General Services Administra- tion (GSA) and the Federal Energy Management Program (FEMP). Tech. Rep. (Septem- ber 2010)

Qureshi, A., Weber, R., Balakrishnan, H., Guttag, J., Maggs, B.: Cutting the electric bill for Internet-scale systems. In: Proc. ACM SIGCOMM, pp. 123–134 (2009)

Rahimi, S.K., Haug, F.S.: Distributed Database Management Systems. John Wiley & Sons, Hoboken (2010)

Rajamani, K., Lefurgy, C.: On Evaluating Request-Distribution Schemes for Saving En- ergy in Server Clusters. In: 2003 IEEE International Symposium on Performance Anal- ysis of Systems and Software (ISPASS 2003), pp. 111–122 (2003)

Rao, L., Liu, X., Ilic, M., Liu, J.: Mec-idc: joint load balancing and power control for distributed internet data centers. In: Proceedings of the 1st ACM/IEEE International Con- ference on Cyber-Physical Systems, pp. 188–197. ACM (2010)

Rao, L., Liu, X., Xie, L., Liu, W.: Minimizing electricity cost: optimization of distributed internet data centers in a multi-electricity-market environment. In: 2010 IEEE Proceed- ings of INFOCOM, pp. 1–9 (2010)

Rossi, D., Metra, C., Nieuwland, A.K., Katoch, A.: Exploiting ECC redundancy to min- imize crosstalk impact. IEEE Design Test Comput. 22(1), 59–70 (2005)

Rossi, D., Metra, C., Nieuwland, A.K., Katoch, A.: New ECC for crosstalk effect mini- mization. IEEE Design Test Comput. 22(4), 340–348 (2005)

Roure, D.D., Goble, C., Stevens, R.: The design and realisation of the Virtual Re-search Environment for social sharing of workflows. Future Generation Computer Sys- tems 25(5), 561–567 (2009)

Sabuncu, M.R., Ramadge, P.J.: Gradient based nonuniform subsampling for information-theoretic alignment methods. In: 26th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society (IEMBS), pp. 1683–1686 (2004)

Sadler, C.M., Martonosi, M.: Data compression algorithms for energy-constrained de- vices in delay tolerant networks. In: SenSys 2006: Proc. of the 4th Int. Conf. on Embed- ded Networked Sensor Systems, pp. 265–278 (2006)

Salomie, I., Cioara, T., Anghel, I., Moldovan, D., Copil, G., Plebani, P.: An Energy Aware Context Model for Green IT Service Centers. In: Maximilien, E.M., Rossi, G., Yuan, S.-T., Ludwig, H., Fantinato, M. (eds.) ICSOC 2010. LNCS, vol. 6568, pp. 169–180. Springer, Heidelberg (2011)

Schmidt, M., Hutchison, B., Lambros, P., Phippen, R.: The Enterprise Ser- vice Bus: making service-oriented architecture real. IBM Systems Journal 44(4), 781–797 (2005), http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm? arnumber=5386706

Schoellhammer, T., Greenstein, B., Osterweil, E., Wimbrow, M., Estrin, D.: Lightweight Temporal Compression of microclimate datasets. In: LCN 2004: Proc. of the 29th Annual IEEE Int. Conf. on Local Computer Networks, pp. 516–524 (2004)

Schubert, L., Wesner, S., Dimitrakos, T.: Secure and Dynamic Virtual Organizations for Business. In: Cunningham, P., Cunningham, M. (eds.) Proc. of the 15th ISPE Interna-tional Conference on Concurrent Engineering (CE 2008), Collaborative Product and Ser-vice Life Cycle Management for a Sustainable World, vol. 2. IOS Press (2005)

Severi, S., Abreu, G., Destino, G., Dardari, D.: Understanding and solving flip-ambiguity in network localization via semidefinite programming. In: GLOBECOM 2009: Proc. of the 28th IEEE Conf. on Global Telecommunications, pp. 3910–3915 (2009)

Shacham, A., et al.: Photonic Network-on-Chip for Future Genera-tions of Chip Multi-Processors. IEEE Transactions on Computers 57(9), 1246–1260 (2008)

Shah, R., Rabaey, J.: Energy aware routing for low energy ad hoc sensor networks. In: 2002 IEEE Wireless Communications and Networking Conference, WCNC 2002, vol. 1, pp. 350–355 (2002)

Sharifi, S., Rosing, T.V.: Accurate direct and indirect on-chip temperature sensing for efficient dynamic thermal management. Trans. Comp.-Aided Des. Integ. Cir. Sys. 29, 1586–1599 (2010)

Sharma, R., Bash, C., Patel, C., Friedrich, R., Chase, J.: Balance of power: Dynamic thermal management for internet data centers. IEEE Internet Computing, 42–49 (2005)

Shen, G., Zhang, Y.Q.: A New Evolutionary Algorithm Using Shadow Price Guided Operators. Applied Soft Computing 11(2), 1983–1992 (2011), doi:10.1016/j.asoc.2010.06.014

Shen, G., Zhang, Y.: A new evolutionary algorithm using shadow price guided operators. Applied Soft Computing 11(2), 1983–1992 (2011)

Shen, G., Zhang, Y.: A shadow price guided genetic algorithm for energy aware task scheduling on cloud computers. In: Advances in Swarm Intelligence, pp. 522–529 (2011)

Shen, G., Zhang, Y.: A shadow price guided genetic algorithm for energy aware task scheduling on cloud computers. In: Proc. of the 3rd International Conference on Swarm Intelligence - (ICSI 2011), pp. 522–529 (2011)

Shen, G., Zhang, Y.-Q.: A Shadow Price Guided Genetic Algorithm for Energy Aware Task Scheduling on Cloud Computers. In: Tan, Y., Shi, Y., Chai, Y., Wang, G. (eds.) ICSI 2011, Part I. LNCS, vol. 6728, pp. 522–529. Springer, Heidelberg (2011), doi:10.1007/978-3-642-21515-5-62

Shen, G., Zhang, Y.Q.: An Evolutionary Linear Programming Algorithm for Solving the Stock Reduction Problem. International Journal of Computer Applications in Technol- ogy (IJCAT) 44(6) (2012)

Shields, M., Taylor, I.: Programming scientific and distributed workflow with Triana ser- vices. Concurrency and Computation: Practice & Experience - Special Issue on “Work- flow in Grid Systems” 18(10), 1–10 (2006)

Skadron, K., Huang, W.: Analytical model for sensor placement on microprocessors. In: 2005 International Conference on Computer Design, pp. 24–27 (2005), http:// ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1524125

Skadron, K., Lee, K.: Using Performance Counters for Runtime Temperature Sens- ing in High-Performance Processors. In: 19th IEEE International Parallel and Dis- tributed Processing Symposium, pp. 232a–232a (2005), http://ieeexplore.ieee. org/lpdocs/epic03/wrapper.htm?arnumber=1420152

Skadron, K., Stan, M., Huang, W., Velusamy, S., Sankaranarayanan, K., Tarjan, D.: Temperature-Aware Microarchitecture. In: ISCA, pp. 2–13 (2003)

Skadron, K., Stan, M., Huang, W., Velusamy, S.: Temperature-aware microarchitecture: Extended discussion and results. University of Virginia, Department of Computer Science (2003), http://scholar.google.com/scholar?q=intitle:Temperature-Aware+Microarchitecture:+Extended+Discussion+and+Results#0

Skadron, K., Stan, M.R., Sankaranarayanan, K., Huang, W., Velusamy, S., Tarjan, D.: Temperature-aware microarchitecture: Modeling and implementation. In: TACO, vol. 1(1), pp. 94–125 (2004)

Song, S., Hwang, K., Kwok, Y.: Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling. IEEE Tran. on Computers 55(6), 703–719 (2006)

Song, S., Hwang, K., Kwok, Y.: Trusted grid computing with security binding and trust integration. J. of Grid Computing 3(1-2), 53–73 (2005)

Sonntag, M., Currle-Linde, N., Go¨rlach, K., Karastoyanova, D.: Towards simulation workflows with BPEL: deriving missing features from GRICOL. In: Proc. of the 21st IASTED International Conference on Modelling and Simulation MS 2010. ACTA Press (2010)

Sonntag, M., Karastoyanova, D., Deelman, E.: BPEL4Pegasus: Combining Business and Scientific Workflows. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) ICSOC 2010. LNCS, vol. 6470, pp. 728–729. Springer, Heidelberg (2010)

Sonntag, M., Karastoyanova, D.: Next generation interactive scientific experimenting based on the workflow technology. In: Proc. of the 21st IASTED International Confer- ence Modelling and Simulation (MS 2010). ACTA Press (2010)

SPEC-CPU 2000, Standard Performance Evaluation Council, Performance Evaluation in the New Millennium, Version 1.1 (2000)

Sridhar, A., Vincenzi, A., Ruggiero, M., Atienza, D., Brunschwiler, T.: 3d-ice: Fast com- pact transient thermal modeling for 3D-ICs with inter-tier liquid cooling. In: Interna- tional Conference on Computer-Aided Design, ICCAD 2010 (2010)

Sridhar, A., Vincenzi, A., Ruggiero, M., Brunschwiler, T., Atienza Alonso, D.: Compact transient thermal model for 3D ICs with liquid cooling via enhanced heat transfer cavity geometries. In: Proceedings of the 16th International Workshop on Thermal Investigations of ICs and Systems (THERMINIC 2010), vol. 1(1), pp. 105–110. IEEE Press, New York (2010)

Sridhara, S.R., Shanbhag, N.R.: Coding for System-on-Chip Networks: A Unified Framework. IEEE Transactions on Very Large Scale Integration (TVLSI) Systems 13(6), 655–667 (2005)

Srinivas, N., Deb, K.: Multiobjective optimization using Nondominated Sorting in Ge- netic Algorithms. IEEE Trans. Evol. Comput. 2(3), 221–248 (1994)

Steinmetz, R., Nahrstedt, K.: Multimedia Systems. Springer, New York (2004)

Systems (NBiS 2009), pp. 424–431 (2009)Apache HTTP Server Version 2.0 Documentation. Available via DIALOG, http:// httpd.apache.org/docs/2.0/en/ (cited August 8, 2011)

Tang, Q., Gupta, S.K.S., Varsamopoulos, G.: Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber- physical approach. IEEE Trans. Parallel Distrib. Syst. 19(11), 1458–1472 (2008)

Tang, Q., Tummala, N., Gupta, S., Schwiebert, L.: Communication scheduling to min- imize thermal effects of implanted biosensor networks in homogeneous tissue. IEEE Transactions on Biomedical Engineering 52(7), 1285–1294 (2005)

Taylor, I.J., Deelman, E., Gannon, D.B.: Workflows for e-Science - Scientific Workflows for Grids. Springer (2007)

Taylor, S., Surridge, M., Laria, G., Ritrovato, P., Schubert, L.: Business Collaborations in Grids: The BREIN Architectural Principals and VO Model. In: Altmann, J., Buyya, R., Rana, O.F. (eds.) GECON 2009. LNCS, vol. 5745, pp. 171–181. Springer, Heidelberg (2009)

Teuscher, C.: Nature-Inspired Interconnects for Self-Assembled Large-Scale Network- on-Chip Designs. Chaos 17(2), 26–106 (2007)

Tian, L., Arslan, T.: A genetic algorithm for energy efficient device scheduling in real- time systems. In: 2003 Congress on Evolutionary Computation, pp. 242–247 (2003), doi:10.1109/CEC.2003.1299581

Tseng, P.: Second–order cone programming relaxation of sensor network localization. SIAM J. Optim. 18(1), 156–185 (2007)

U.S. Energy Information Administration: Independent Statistic and Analysis, Re- newable Energy Consumption and Electricity Preliminary 2006 Statistics (2006), http://www.eia.doe.gov/cneaf/solar.renewables/page/prelim_trends/ rea_prereport.html (accessed July 2011)

Vecchio, M., Lo´pez-Valcarce, R., Marcelloni, F.: A study on the application of different two-objective evolutionary algorithms to the node localization problem in wireless sen- sor networks. In: ISDA 2011: Proc. of the 11th IEEE Int. Conf. on Intelligent Systems Design and Applications, pp. 1008–1013 (2011)

Vecchio, M., Lo´pez-Valcarce, R., Marcelloni, F.: A two-objective evolutionary approach based on topological constraints for node localization in wireless sensor networks. Ap- plied Soft Computing 12(7), 1891–1901 (2012), doi:10.1016/j.asoc.2011.03.012

Velusamy, S., Huang, W., Lach, J., Stan, M.R., Skadron, K.: Monitoring temperature in fpga based socs. In: ICCD, pp. 634–640 (2005)

Viswanath, R., Wakharkar, V., Watwe, A., Lebonheur, V.: Thermal performance chal-lenges from silicon to systems. Intel Technology Journal Q3, 1–16 (2000)

Wang, L., Von Laszewski, G., Dayal, J., He, X., Furlani, T.R.: Thermal Aware Workload Scheduling with Backfilling for Green Data Centers. In: the 28th IEEE Intl. Conf. on Performance Computing and Communications, pp. 289–296 (2009), doi:10.1109/PCCC.2009.5403821

Wang, L., von Laszewski, G., Dayal, J., Wang, F.: Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with dvfs. In: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, pp. 368–377 (2010)

Wang, L., Von Laszewski, G., Dayal, J., Wang, F.: Towards Energy Aware Schedul- ing for Precedence Constrained Parallel Tasks in a Cluster with DVFS. In: 2010 10th IEEE/ACM Intl. Conf. on Cluster, Cloud and Grid Computing (CCGrid), pp. 368–377 (2010), doi:10.1109/CCGRID.2010.19

Wang, Z., Zheng, S., Ye, Y., Boyd, S.: Further relaxations of the semidefinite program- ming approach to sensor network localization. SIAM J. Optim. 19(2), 655–673 (2008)

Watts, D.J.: Small Worlds: The Dynamics of Networks between Order and Randomness pp. xvi–262. Princeton University Press (1999)

Weighted Round Robin (WRR), Available via DIALOG, http://www. linuxvirtualserver.org/docs/scheduling.html (cited December 7, 2011)

Whitley, D., Rana, S., Heckendorn, R.: The island model genetic algorithm: On separa- bility, population size and convergence. Journal of Computing and Information Technol- ogy 7, 33–47 (1998)

Wieland, M., Gorlach, K., Schumm, D., Leymann, F.: Towards reference passing in web service and workflow-based applications. In: Proc. of 2009 IEEE International Enterprise Distributed Object Computing Conference, pp. 109–118 (2009), http://ieeexplore. ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5277706

Wu, C.-C., Sun, R.-Y.: An integrated security-aware job scheduling strategy for large- scale computational grids. Future Generation Computer Systems 26(2), 198–206 (2010)

Xhafa, F., Abraham, A.: Computational models and heuristic methods for grid schedul- ing problems. Future Generation Computer Systems 26, 608–621 (2010)

Xhafa, F., Carretero, J., Abraham, A.: Genetic algorithm based schedulers for grid com- puting systems. Int. J. of Innovative Computing, Information and Control 3(5), 1053– 1071 (2007)

Xhafa, F., Carretero, J., Barolli, L., Durresi, A.: Requirements for an event-based sim- ulation package for grid systems. Journal of Interconnection Networks 8(2), 163–178 (2007)

Xiang, Y., Chantem, T., Dick, R.P., Hu, X.S., Shang, L.: System-level reliability model- ing for mpsocs. In: Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, CODES/ISSS 2010, pp. 297– 306 (2010), http://doi.acm.org/10.1145/1878961.1879013

Xie, Y., Wang, Z., Wei, S.: An efficient algorithm for non-preemptive periodic task scheduling under energy constraints. In: 6th Intl. Conf. on ASIC, pp. 128–131 (2005), doi:10.1109/ICASIC.2005.1611282

Xiong, Z., Liveris, A., Cheng, S.: Distributed source coding for sensor networks. IEEE Signal Process. Mag. 21(5), 80–94 (2004)

Xue, F., Sanderson, A., Graves, R.: Multi-objective routing in wireless sensor networks with a differential evolution algorithm. In: Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control, ICNSC 2006, pp. 880–885 (2006)

Yang, K., Liu, X.: Improving the Performance of the Pareto Fitness Genetic Algorithm for Multi-Objective Discrete Optimization. In: Intl. Symposium Computational Intelli- gence and Design, vol. 2, pp. 394–397 (2008)

Yang, T., Mino, G., Spaho, E., Barolli, L., Durresi, A., Xhafa, F.: A Simulation System for Multi Mobile Events in Wireless Sensor Networks. In: The 25th IEEE International Conference on Advanced Information Networking and Applications (AINA 2011), pp. 411–418 (2011)

Yang, Y., Xiong, N., Aikebaier, A., Enokido, T., Takizawa, M.: Minimizing Power Con- sumption with Performance Efficiency Constraint in Web Server Clusters. In: The 12th International Conference on Network-Based Information Systems (NBiS 2009), pp. 45–51 (2009)

Yeo, I., Liu, C.C., Kim, E.J.: Predictive dynamic thermal management for multicore systems. In: DAC, pp. 734–739 (June 2008)

Yeo, I., Liu, C.C., Kim, E.J.: Predictive dynamic thermal management for multicore sys- tems. In: Proceedings of the 45th Annual Design Automation Conference, DAC 2008, pp. 734–739. ACM, New York (2008), http://doi.acm.org/10.1145/1391469. 1391658

Yu, G., Kouvels, P.: On min-max optimization of a collection of classical discrete optimization problems. Optimization Theory and Applications 98(1), 221–242 (1998)

Yu, J., Buyya, R., Tham, C.: Cost-based scheduling of scientific workflow application on utility grids. In: Proceedings of the First International Conference on e-Science and Grid Computing, pp. 140–147. IEEE Computer Society (2005)

Yuen, S., Chow, C.: A Genetic Algorithm That Adaptively Mutates and Never Revisits. IEEE Trans. Evolutionary Computation 13(2), 454–472 (2009)

Zhang, L., Li, K., Zhang, Y.: Green task scheduling algorithms with speeds optimization on heterogeneous cloud servers. In: Proceedings of the 2010 Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing, pp. 76–80. IEEE Computer Society (2010)

Zhang, L., Zhou, Q.: CCOA: Cloud Computing Open Architecture. In: IEEE International Conference on Web Services, pp. 607–616 (2009)

Zhao, D., Wang, Y.: SD-MAC: Design and Synthesis of A Hardware-Efficient Collision-Free QoS-Aware MAC Protocol for Wireless Network-on-Chip. IEEE Transactions on Computers 57(9), 1230–1245 (2008)

Zhou, Y., et al.: Design and Fabrication of Microheaters for Localized Carbon Nanotube Growth. In: Proc. of IEEE Conference on Nanotechnology, pp. 452–455 (2008)

Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: Empirical results. IEEE Trans. Evol. Comput. 8(2), 173–195 (2000)

Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength Pareto evolution- ary algorithm for multiobjective optimization. In: Giannakoglou, K., et al. (eds.) Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001), pp. 95–100. Int. Center for Numerical Methods in Engineering (CIMNE), Barcelona (2002)

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, G.V.: Performance assessment of multiobjective optimizers: An analysis and review. IEEE Trans. Evol. Comput. 7, 117–132 (2002)

Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans. Evol. Comput. 3(4), 257–271 (1999)

Zomaya, A.Y.: Energy-aware scheduling and resource allocation for large-scale distributed systems. In: 11th IEEE International Conference on High Performance Computing and Communications (HPCC), Seoul, Korea (2009)

KOMPUTASI HIJAU
(Green Computing)

Dr. Budi Raharjo, S.Kom, M.Kom.,MM.

BIODATA PENULIS

Dr. Budi Raharjo, S.Kom, M.Kom, MM lahir di Semarang, tanggal 22 Februari 1985. Beliau adalah Alumni dari Universitas Bina Nusantara (BINUS University) Jakarta dan juga alumni Universitas Kristen Satya wacana (UKSW) Salatiga. Dr. Budi Raharjo telah menjadi Dosen pada Universitas STEKOM pada mata kuliah Kepemimpinan (Leadership), mata kuliah Pengantar Akuntansi, Manajemen Proses, Manajemen Akuntansi dan Manajemen Resiko Bisnis. Selain sebagai dosen Universitas STEKOM, Dr. Budi Raharjo, M.Kom, MM juga mempunyai bisnis sendiri dalam bidang perhotelan dan juga sebagai wirausaha dalam bidang pemasok unggas (ayam) beku, ke berbagai kota besar, khususnya Jakarta dan sekitarnya.

Pengalaman beliau berwirausaha menjadi bekal utama dalam penulisan buku ajar yang diterbitkan oleh Yayasan Prima Agus Teknik (YPAT) Semarang. Oleh sebab itu bukunya berisi langkah langkah praktis yang mudah diikuti oleh para mahasiswa, saat mahasiswa mengikuti proses perkuliahan pada Universitas Sains dan Teknologi Komputer (Universitas STEKOM). Jabatan struktural yang di embannya saat ini adalah Wakil Rektor 1 (Akademik) Universitas STEKOM Semarang.

YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-623-8120-13-0 (PDF)

Dr. Budi Raharjo, S.Kom, M.Kom.,MM.

KOMPUTASI HIJAU
(Green Computing)

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id