PENJADWALAN KULIAH MENGGUNAKAN METODE CONSTRAINTS PROGRAMMING DAN SIMULATED ANNEALING

Abdul Rochman
Jurusan Teknik Informatika, Fakultas Teknologi Industri,Universitas Trisakti
Jl. Kyai Tapa No. 1, Jakarta 11440
Telp. (021) 5663232 ext. 8436
E-mail: bbc_abdul92@yahoo.com

ABSTRAK
Aplikasi Penjadwalan Kuliah pada umumnya melakukan dua tahapan komputasi. Tahapan pertama menghasilkan suatu jadwal awal yang telah memenuhi hard constraints. Jadwal awal ini selanjutnya digunakan sebagai masukan pada tahap kedua, yaitu suatu komputasi yang melakukan perbaikan dalam penurunan jumlah pelanggaran soft contraints. Dalam penelitian ini telah dikembangkan aplikasi penjadwalan kuliah yang menerapkan constraints programming pada tahapan pertama dan menerapkan simulated annealing pada tahapan kedua. Hasil uji coba memperlihatkan terjadinya penurunan pelanggaran syarat sebesar 23.4% dan terjadi penurunan jumlah kelas yang tidak teralokasi sebesar 34.5% pada jadwal akhir yang dihasilkan.

Kata Kunci: hard constraints, soft constraints, constraints programming, simulated annealing.

1. PENDAHULUAN

Penjadwalan dapat didefinisikan sebagai sebuah permasalahan yang memiliki empat parameter: himpuan berhingga waktu (T), himpunan berhingga sumber daya (R), himpunan berhingga pertemuan (M), dan himpunan berhingga batasan (C). Dalan hal ini, permasalahan yang dimaksud adalah pengalokasian waktu dan sumber daya terhadap suatu pertemuan untuk memenuhi batasan-batasan sebanyak mungkin (Burke, deWera dan Kingstone 2004).

Penjadwalan kuliah termasuk sebuah permasalahan NP-*hard*, sulit diselesaikan dengan menggunakan metode yang konvensional dan waktu komputasi yang dibutuhkan untuk mendapatkan solusi yang optimal meningkat eksponensial seiring besarnya permasalahan. Penjadwalan kuliah merupakan kegiatan reguler dalam suatu institusi pengajaran. Kegiatan ini berusaha mengalokasikan sumber daya yang dimililiki institusi (seperti tenaga pengajar dan ruangan) kedalam kegiatan-kegiatan belajar-mengajar (seperti perkuliahan dan praktikum) dalam sebuah kalender mingguan yang memenuhi sejumlah batasan. Batasan-batasan ini biasanya dikelompokkan kedalam dua kelompok: *hard constraints* dan *soft constraints*. Hard constraints memiliki prioritas lebih tinggi dari soft constraints. Tujuan dari permasalahan ini adalah untuk memenuhi *hard constraints* dan meminimalisasi pelanggaran pada *soft constrains*.

Beragam metode telah diusulkan untuk menyelesaikan permasalahan penjadwalan seperti permasalahan *graph-coloring* dan *constraints programming* (Duong 2004) (Zang 2005). Terdapat juga beragam metode meta-heuristic seperti simulated annealing, tabu search dan algoritme genetik (All-Milli 2011). Metode-metode ini berasal dari sejumlah disiplin ilmu seperti Riset Operasi, *Artificial Intelligence* dan *Computational Intelligences*.

Tulisan ini disusun dengan urutan sebagai berikut: Penjelasan *constraint programming* dan batasan-batasan permasalahan penjadwalan dalam Jurusan Teknik Informatika Universitas Trisakti terdapat di bagian ke dua. Bagian ke tiga membahas algoritme simulated annealing. Bagian ke empat berisi pembahasan implementasi aplikasi penjadwalan menggunakan metode constraint programming dan simulated annealing dan uji coba aplikasi. Bagian ke lima memuat kesimpulan.

2. CONSTRAINT PROGRAMMING

Constraint satisfaction problem (CSP) secara formal dapat didefinisikan sebagai permasalahan yang direpresentasikan dalam bentuk himpunan variabel dan syarat-syarat (*constraints*) (Russel 2003). Setiap variabel memiliki ranah yang berisikan nilai-nilai yang mungkin diberikan kepadanya. Sedangkan setiap syarat mencakup beberapa variabel dan membatasi kombinasi nilai-nilai yang dapat diberikan kepada pasangan variabel tadi. Ide ini juga diangkat sebagai suatu paradigma dalam pemrograman, dan dikenal sebagai *constraint programming*.

Syarat dapat dikategorikan berdasarkan sifatnya kedalam dua kategori, yaitu *Absolute constraint* atau *hard constraint*, merupakan batasan-batasan yang harus dipenuhi. Apabila terjadi pelanggaran terhadap syarat jenis ini, solusi dinyatakan tidak *valid*. Kategori yang kedua, yaitu *preference constraint* atau *soft constraint*, merupakan syarat-syarat yang ingin dicapai. Pemenuhan dari syarat jenis ini akan memperbaiki nilai solusi fungsi tujuan. Namun, pelanggaran dari batasan ini tidak menyatakan solusi

tidak *valid*, hanya memperburuk nilai solusi dalam pemenuhan fungsi tujuan.

Bentuk *hard constraints* yang ditemukan dalam penjadwalan kuliah pada Jurusan Teknik Informatika, FTI USAKTI adalah :

C1. Seorang dosen tidak dapat berada pada dua ruangan berbeda pada waktu yang sama. Apabila seorang dosen telah dijadwalkan untuk mengajar suatu matakuliah pada waktu tertentu, dosen tersebut tidak dapat dijadwalkan lagi untuk matakuliah apapun pada waktu yang sama.

C2. Sebuah ruangan hanya dapat dipergunakan untuk satu kelas pada satu waktu tertentu. Apabila suatu slot ruang waktu telah terjadwal untuk suatu kelas, kelas lainnya tidak dapat menggunakan ruangan tersebut lagi pada waktu yang sama.

C3. Kapasitas ruangan harus sama dengan atau lebih besar dari kapasitas kelas yang akan diadakan. Apabila suatu kelas direncanakan memiliki kapasits *n*, ruangan yang diberikan untuk kelas tersebut harus memiliki kapasitas minimum sama dengan *n*.

C4. Matakuliah-matakuliah yang dimaksudkan untuk suatu semester tidak dapat diadakan pada waktu yang sama. Apabila matakuliah A untuk semester *n* diadakan pada waktu tertentu, matakuliah-matakuliah lain yang dimaksudkan untuk semester *n* pula tidak dapat diadakan pada waktu tersebut, kecuali kelas paralel untuk matakuliah A. Hal ini dimaksudkan agar mahasiswa/i dapat mengambil setiap matakuliah yang dimaksudkan untuk semester tersebut.

Sedangkan, *soft constraints* yang ditemukan pada permasalahan penjadwalan terdiri dari :

C5. Sebaran waktu ajar dosen. Diusahakan agar seorang dosen tidak mengajar lebih dari *n* buah matakuliah per hari. Pada penelitian ini, nilai *n* tersebut akan ditentukan sebanyak 2 matakuliah perhari.

C6. Sebaran waktu belajar mahasiswa. Matakuliah-matakuliah yang dimaksudkan untuk semester tertentu diusahakan tidak melebihi *n* buah matakuliah per hari. Untuk penelitian ini akan digunakan nilai *n* sebesar tiga matakuliah. Syarat ini bersama dengan syarat pertama dimaksudkan agar kegiatan belajar menjadi lebih efektif, karena dengan pembatasan diatas diharapkan dosen dapat mempersiapkan dan mengajarkan suatu matakuliah secara lebih baik. Dilain pihak, diharapkan pula bagi mahasiswa dapat menerima pengajaran secara lebih baik.

C7. Pengusahaan pengadaan kelas-kelas suatu matakuliah secara bersamaan. Apabila suatu matakuliah mengadakan lebih dari satu kelas, akan diusahakan agar setiap kelasnya diadakan pada waktu yang sama.

C8. Pengusahaan agar setiap kelas yang diadakan memilki slot ruang waktu dan seorang pengajar.

3. SIMULATED ANNEALING

Simulated Annealing (SA) adalah suatu teknik pencarian berdasarkan probabilistik. Prosesnya dimulai dengan membuat inisialisasi solusi awal secara acak (S_0). Setiap iterasi, algoritme SA mengganti solusi saat ini (S) dengan sebuah solusi baru (*S). Solusi baru ini dipilih secara acak dari tetangga S, *S∈N(S)=tetangga S, menggunakan fungsi probabilitas exp(δ/t). Suhu t biasanya diinisialisasi dengan suhu tinggi dan berkurang hingga akhir proses menggunakan suatu fungsi penurunan suhu α. Algoritme simulated annealing dapat dilihat pada gambar 1.

```
Pilih sebuah solusi awal S0
Pilih sebuah suhu awal t0 > 0
Pilih sebuah fungsi penurunan suhu α
S = S0; t = t0
repeat
  repeat
     Pilih secara acak S'∈ N(S)
     δ = f(S')-f(S)
     if(δ <0) then S = S'
     else
        hasilkan secara acak x∈[0,1]
        if x < exp(-δ/t) then S = S'
        endif
     endif
   until cacah_iterasi = nrep
   t= α(t)
until kondisi_stop = true
```

Gambar 1. Algoritme Simulated Annealing

4. IMPLEMENTASI DAN UJI COBA

Sistem penjadwalan kuliah yang dikembangkan melakukan dua proses komputasi: pengembangan jadwal awal, menerapkan constraint programming untuk menghasilkan suatu jadwal yang telah memenuhi semua hard constraints; dan pengoptimalan jadwal, menerapkan metode simulated annealing untuk meningkatkan kualitas dari jadwal yang dihasilkan.

Proses pengembangan jadwal awal dapat dipecah menjadi dua subproses, yaitu subproses pengalokasian berdasarkan waktu dosen dan subproses pengalokasian sisa kelas. Urutan aktivitas dari kedua subproses dapat dilihat pada gambar 2 dan gambar 3 yang terdapat pada halaman terakhir dari tulisan ini. Begitu pula dengan urutan aktivitas dari proses pengoptimalan jadwal dapat dilihat pada gambar 4.

Dalam implementasi, suhu awal diinisialiasi dengan nilai 10000 ^{o}C, suhu akhir sebesar 0,005 ^{o}C, dan fungsi penurunan suhu $t = \alpha \; x \; t$ [1]. Nilai α didapat dari rumus: $\alpha = 1 - (\ln(T_0) - \ln(T_f)) / N_{move}$, dimana T_0 merupakan suhu awal, T_f merupakan suhu akhir dan N_{move} adalah jumlah perulangan yang diinginkan.

Dalam pengujian, baik modul pengembang jadwal awal maupun modul pengoptimalan jadwal,

menggunakan data-data riil yang pernah dipergunakan di jurusan teknik Informatika Trisakti pada semester gasal 2004/2005. Data-data tersebut memiliki keterangan sebagai berikut:

i. Terdapat 25 orang dosen yang dapat mengajar pada semester tersebut.

ii. Terdapat 44 matakuliah yang akan ditawarkan, dimana masing-masing matakuliah akan diadakan antara satu hingga tiga kelas, dengan total 73 kelas yang akan diadakan.

iii. Terdapat 578 mahasiswa aktif.

iv. Terdapat empat ruangan kelas, masing-masing memiliki kapasitas: 30, 30, 40 dan 60 siswa.

v. Setiap dosen yang aktif memiliki satu hingga empat matakuliah keahlian.

vi. Waktu perkuliahan dari hari Senin hingga hari Sabtu mulai dari jam 7.30 hingga jam 17.30.

vii. Ketersediaan waktu dosen amat beragam, dari yang memiliki hanya satu kesediaan waktu hingga sembilan belas.

Uji coba dilakukan pada komputer dengan spesifikasi perangkat keras dan perangkat lunak: Intel Pentium III, 392,668 KB RAM, Sistem Operasi Windows 2000 Advanced Server, Java 2 SDK, SE v1.4.2, Java 2 Runtime Environment, SE v1.4.2, NetBeans IDE 3.5 dan basis data Microsoft Access 2000.

Hasil eksekusi modul pengembang jadwal sebanyak lima kali memperlihatkan bahwa banyaknya kelas yang tidak terjadwal sebanyak 29 kelas(39,7%), tidak ada dosen yang mengajar lebih satu kelas dalam satu hari (kolom distr. dosen) dan terbentuk 17 kelas paralel. Jumlah pelanggaran *soft constraints* yang terjadi sebanyak 47. Waktu eksekusi rata-rata adalah 488 mili detik.

Tabe 1. Hasil Eksekusi Pengembangan Jadwal

No	*Nilai Jadwal*	*Kelas Paralel*	*Distr. Dosen*	*TakTer-alokasi*	*Waktu (ms)*
1	47	17	1	29	478
2	47	17	1	29	507
3	47	17	1	29	510
4	47	17	1	29	438
5	47	17	1	29	508

Setelah didapatkan jadwal awal, selanjutnya dilakukan proses optimasi menggunakan metode simulated annealing. Dari hasil uji coba, seperti yang terlihat pada tabel 2, nilai rata-rata pelanggaran sebesar 36.4, yang berarti terjadi penurunan jumlah pelanggaran sebesar 23.4%. Distribusi beban mengajar setiap dosen memenuhi keinginan, yaitu sebanyak dua matakuliah dan pelaksanaannya dilakukan dalam satu hari. Jumlah kelas yang tidak terjadwal sebanyak 19 kelas, yang berarti terjadi penurunan sebesar 34.5%.

Tabel 2. Hasil Eksekusi Modul Optimasi Jadwal

No	*Nilai Jadwal*	*Kelas Paralel*	*Distr. Dosen*	*TakTer-alokasi*	*Waktu (ms)*
1	36	15	2	19	3033
2	36	15	2	19	3192
3	36	15	2	19	2911
4	36	15	2	19	2971
5	38	17	2	19	3595

5. KESIMPULAN

Jadwal awal yang didapat dari metode *constraint programing* memperlihatkan hasil yang cukup baik, hal ini terlihat dari presentase kelas yang tidak terjadwal sebesar (39,7%). Waktu eksekusi yang dibutuhkan untuk membentuk jadwal awal ini relatif cepat yaitu sebesar 488 mili detik.

Proses optimasi yang dilakukan menggunakan metode simulated annealing memperlihatkan perbaikan jadwal akhir yang dihasilkan. Terjadi penurunan jumlah pelanggaran sebesar 23.4%, peningkatan distribusi beban mengajar dosen dari satu matakuliah menjadi dua matakuliah, penurunan jumlah kelas yang tidak teralokasi sebesar 34.5%.

PUSTAKA

All-Milli, N. R. (2011). Hybrid Genetic Algorithms With Simulated Annealing for University Course Timetabling Problems. *Journal of Theoretical and Applied Information Technology*. 29(2), 100-106.

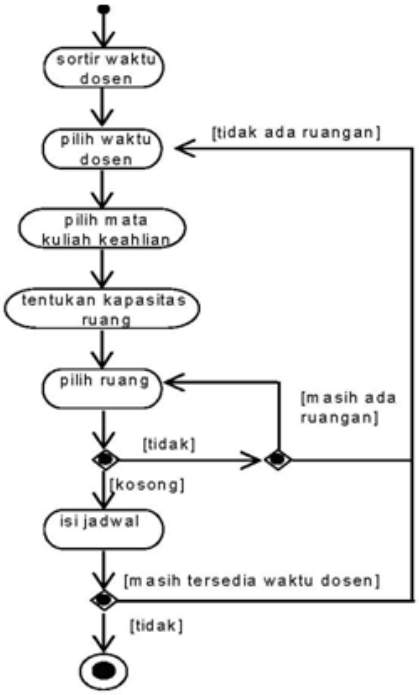
Burke, E. K., de Werra, dan Kingstone, H. (2004). Applications to Timetabling, in: J.Gross dan J.Yellan. *The Hand Book of Graph Theory*, Chapman Hall/CRC Press, 445-474.

Duong, T. A., dan Lam, K. H. (2004). *Combining Constraint Programming and Simulated Annealing On University Exam Timetabling*, Hanoi.

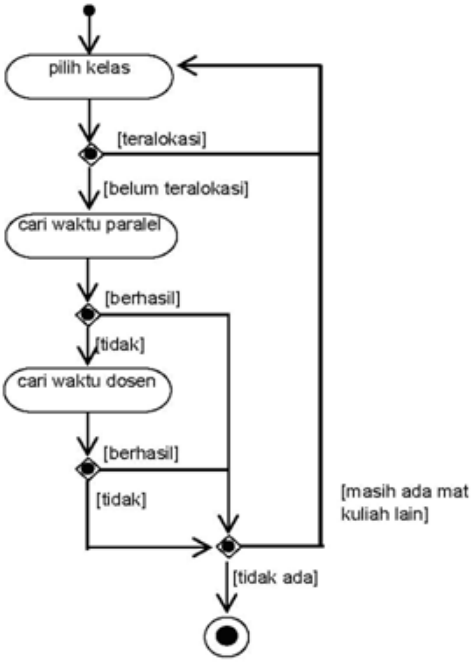
Nandhini, N., dan Kanmani, S. (2009). A Survey of Simulated Annealing Methodology for University Course Timetabling. *International Journal of Recent Trends in Engineering*, 1(2), 255-257.

Russel, S., dan Norvig, P. (2003). *Artificial Intelligence A Modern Approach*. New Jersey:Prentice Hall, 115-150.

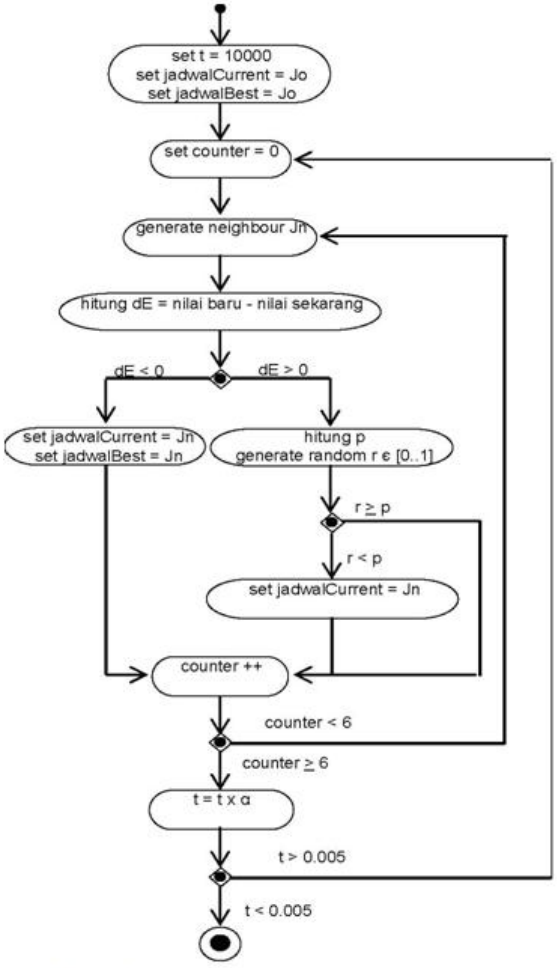
Zang, L., dan Lau, S. (2005). Constructing University Timetable Using Constraint Satisfaction Programming Approach. *International Conference on Computational Intelligence for Modeling, Control and Automations, and International Conference on Intelligent Agent, Web Technology and Internet Commerce*. 55-60.



Gambar 2. Diagram Aktivitas Pengalokasian Berdasarkan Waktu Dosen.



Gambar 3. Diagram Aktivitas Pengalokasian Sisa Kelas.



Gambar 4. Diagram Aktivitas Pengoptimalan Jadwal.