

PENERAPAN TEKNIK *INTELLIGENT SEARCH* PADA PENJADWALAN KULIAH DI PERGURUAN TINGGI

Anbulagan dan Robert Setiadi
Jurusan Teknik Informatika, Fakultas Ilmu Komputer, Universitas Bina Nusantara
Jl. Hang Lekir I No. 6, Jakarta 12120, Indonesia
e-mail : anbulagan@binus.ac.id

ABSTRAK

Artikel ini akan mencoba membahas pemecahan masalah penjadwalan kuliah dengan pendekatan ilmu *Intelegensia Semu (Artificial Intelligence)*, yakni dengan menggunakan *Constraint Satisfaction Problem*. Penulis telah merancang dan menguji sebuah teknik baru yang merupakan pengembangan lebih lanjut dari teknik pencarian solusi dengan *intelligent search* yang dikombinasikan dengan algoritma *Smart Backtracking*.

Algoritma yang kami kembangkan ini telah dicoba dengan sejumlah studi kasus berskala kecil (7 dosen, 7 matakuliah, 23 kelas, 2 ruang, 35 jam perkuliahan tiap minggu, dan lebih dari 1380 mahasiswa) dan menghasilkan output yang diinginkan dalam waktu yang sangat singkat. Percobaan dengan *real data* (1198 dosen, 1457 matakuliah, 2311 kelas, 122 ruang, 40 jam perkuliahan tiap minggu, dan lebih dari 20000 mahasiswa) telah menghasilkan solusi yang baik, meskipun tidak dapat mencapai solusi 100% lengkap. Sejumlah *constraint*, terutama yang berkaitan dengan dosen dan mahasiswa kelas paket seringkali sulit dipenuhi karena adanya sejumlah kelas yang merupakan gabungan dari beberapa paket (dapat mencapai 12).

Kata kunci : *Timetabling, Constraint Satisfaction Problem (CSP), Artificial Intelligence (AI), Intelligent Search, Smart Backtracking*.

1. PENDAHULUAN

Masalah penjadwalan kuliah merupakan masalah sangat kompleks yang hingga saat ini masih merupakan sebuah topik yang banyak dibahas dalam berbagai tesis, disertasi, dan karya ilmiah di seluruh penjuru dunia. Inti dari penjadwalan kuliah adalah bagaimana menjadwalkan sejumlah komponen yang terdiri atas mahasiswa, dosen, ruang, dan waktu dengan sejumlah batasan dan syarat (*constraint*) tertentu.

Sejumlah algoritma dikembangkan untuk menyelesaikan problem ini sesuai dengan kebutuhan instansi yang memerlukannya. Keanekaragaman ini tidak dapat terlepas dari kenyataan bahwa setiap perguruan tinggi

memiliki sejumlah komponen dan kendalanya sendiri dalam menyusun jadwal kuliah.

Penelitian ini memfokuskan diri pada penyelesaian masalah penjadwalan kuliah di Universitas Bina Nusantara. Sejumlah masalah dan kendala terjadi dalam penyusunan jadwal kuliah di universitas ini. Sejumlah kendala yang perlu digarisbawahi adalah tidak adanya pemisahan gedung kampus antar fakultas (mahasiswa dari semua fakultas dan jurusan melakukan kegiatan perkuliahan di kampus yang sama), serta adanya sistem paket kuliah yang memungkinkan mahasiswa dari semester I sampai 5 tidak perlu menempuh proses KRS dalam penentuan kuliah.

Tujuan utama dari adanya sistem paket adalah untuk mempermudah mahasiswa dalam menjalani masa studinya. Sistem ini menjamin setiap mahasiswa mendapatkan kelas yang tepat sesuai dengan matakuliah yang boleh diambilnya dengan urutan tertentu. Tidak perlu lagi adanya antrian panjang untuk memperebutkan kelas karena semuanya telah diatur dengan otomatis oleh sistem.

Perancangan sebuah sistem yang mampu menangani hal di atas secara otomatis bukan merupakan suatu hal yang mudah. Sejumlah penelitian telah dilakukan untuk menyelesaikan masalah penjadwalan ini tanpa adanya campur tangan koreksi manual dari manusia, namun hingga kini belum terdapat sebuah penyelesaian yang benar-benar memuaskan.

Sebuah sistem yang baik harus mampu menangani dan menganalisa kondisi mahasiswa, dosen, matakuliah, ruang kelas, dan keterkaitannya, serta menghasilkan sebuah solusi yang valid dalam waktu proses yang tidak terlalu lama. Apabila sebuah sistem ternyata menghasilkan bahwa seorang dosen atau mahasiswa harus berada di dua tempat sekaligus pada waktu yang sama, maka jadwal kuliah tersebut dapat dinyatakan tidak valid.

Pada umumnya, masalah penjadwalan kuliah yang terdapat di sejumlah instansi lain di seluruh dunia lebih condong pada masalah penanganan waktu proses yang cenderung eksponensial. Akan tetapi, hal tersebut tidak dapat diterapkan pada kasus Universitas Bina Nusantara. Sejumlah uji coba yang telah dilakukan menunjukkan bahwa data input dari sistem memang memiliki sejumlah ketidaksempurnaan yang menyebabkan jadwal sempurna tidak dapat dihasilkan.

Makalah diterima [16 Oktober 2001]. Revisi Akhir [1 Mei 2002]

Pada bagian-bagian berikut, kami akan menguraikan constraint yang ada dalam pembuatan penjadwalan kuliah pada bagian 2. Selanjutnya pada bagian 3, kami akan menyajikan algoritma smart backtracking yang diikuti oleh hasil yang diperoleh dengan menggunakan algoritma tersebut pada bagian 4.

2. RINGKASAN RISET DI BIDANG TIMETABLING

Ukuran keberhasilan sebuah algoritma penjadwalan kuliah harus ditinjau dari keseluruhan aspek. Implementasi algoritma ini pada institusi besar dan institusi kecil pasti memiliki perbedaan, baik dalam jumlah data maupun banyaknya *constraint*. Pada tabel 1., kami sajikan daftar sejumlah referensi penelitian *timetabling*, algoritma yang dipakai, jumlah komponen, dan hasil yang diperoleh.

Tidak satupun dari teknik yang disebutkan di tabel 1. dapat dikatakan sempurna. Penelitian tentang masalah *timetabling* masih terus dilakukan untuk mencari teknik penyelesaian yang lebih baik. Sejak tahun 1995, *International Conference on Practice and Theory of Automated Timetabling (ICPTAT)* telah diselenggarakan setiap tahunnya untuk membahas perkembangan penelitian masalah *automated timetabling* beserta teknik-teknik yang berbasis kecerdasan komputasional. Perbedaan kebutuhan dan permasalahan yang dihadapi oleh masing-masing institusi menyulitkan adanya sebuah solusi global untuk mengatasi berbagai permasalahan *timetabling* di seluruh institusi di dunia.

3. CONSTRAINT PADA PENJADWALAN KULIAH DI UNIVERSITAS BINA NUSANTARA

Dalam pembuatan jadwal kuliah, analisa constraint sangat diperlukan untuk dapat menyelesaikan masalah yang dihadapi dengan tepat. Dari hasil analisa constraint yang dilakukan pada sistem penjadwalan kuliah di Universitas Bina Nusantara, maka ditemukan sejumlah constraint yang dapat dikelompokkan menjadi *hard constraint* dan *soft constraint*.

3.1. Hard Constraint

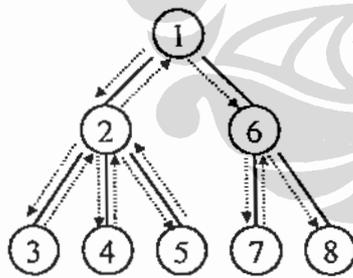
Hard constraint didefinisikan sebagai *constraint* wajib yang harus dipenuhi dalam proses perhitungan algoritma. Sebuah solusi hanya dapat dikatakan sah dan *valid* apabila dalam solusi tersebut sama sekali tidak ada *hard constraint* yang terlanggar.

Berikut adalah daftar *hard constraint* dalam masalah penjadwalan kuliah di Universitas Bina Nusantara beserta penjelasannya :

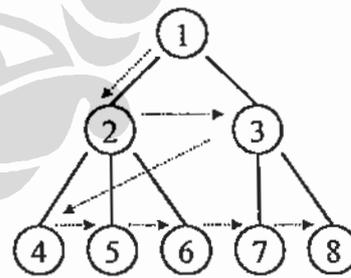
- a) Jadwal dosen tidak boleh bentrok. Pada bagian selanjutnya constraint ini akan diberi kode DB (Jadwal Dosen Bentrok).
- b) Jadwal kuliah mahasiswa paket tidak boleh bentrok. Kode : MB (Jadwal Mahasiswa Paket Bentrok).
- c) Ruang yang digunakan tidak boleh bentrok. Kode : RB (Ruang Bentrok).
- d) Dosen harus mengajar matakuliah yang sesuai dengan ketentuan Ketua Jurusan. Kode : MT (Matakuliah Dosen). Pada tiap awal semester, semua Ketua Jurusan akan menentukan daftar matakuliah yang boleh diajar oleh seorang dosen. Ketentuan ini dibuat berdasarkan spesialisasi dosen serta hasil evaluasi dosen yang tercermin dalam nilai IKAD (Indeks Kinerja Akademik Dosen) yang dicapai masing-masing dosen. Pengontrolan matakuliah ini dilakukan untuk menjaga mutu pendidikan di universitas tersebut.
- e) Dosen harus mendapatkan jadwal mengajar sesuai dengan waktu kesediaan mengajarnya. Kode : WD (Waktu Dosen). Pada akhir semester, semua dosen diberi kesempatan untuk mengisi form kesediaan mengajar dengan maksud agar dosen dapat menyusun jadwal kegiatan agar tidak berbenturan dengan kegiatan lainnya. Hal ini dipandang perlu dilakukan mengingat sebagian besar dosen dari universitas ini adalah dosen honorer.
- f) Dosen tidak boleh mendapatkan jadwal mengajar melebihi jatah SKS yang telah ditentukan. Kode : SD (SKS Dosen). Seperti halnya penentuan matakuliah, Ketua Jurusan juga menentukan jumlah maksimal SKS dosen per minggu.
- g) Seluruh kelas yang dibuka harus mendapatkan waktu dan ruangan perkuliahan. Kode : WR (Waktu dan Ruang). Sebuah kelas tidak dapat berjalan tanpa ruangan dan waktu perkuliahan. Apabila kedua kebutuhan ini tidak dapat dipenuhi, berarti kelas tidak dapat dibuka untuk semester ini.
- h) Jumlah mahasiswa dalam satu kelas tidak boleh melebihi kapasitas ruangan. Kode : KR (Kapasitas Ruang).
- i) Kelas matakuliah yang membutuhkan ruangan khusus harus dijadwalkan di ruangan yang tepat. Kode : RK (Ruang Khusus). Beberapa matakuliah membutuhkan sarana dan prasarana khusus yang tidak terdapat pada kelas-kelas perkuliahan pada umumnya. Hal ini terjadi antara lain pada sejumlah matakuliah mahasiswa jurusan Desain Komunikasi Visual. Matakuliah khusus ini harus dialokasikan pada ruangan yang khusus pula agar kegiatan perkuliahan dapat berjalan dengan baik.

Tabel 1. Ringkasan Beberapa Hasil Penelitian di Bidang Timetabling

Author dan Tahun	Algoritma	Institusi (exam, mhs, period)	Hasil
Foxley & Lockyer (1968) [2]	Sequential Approach	Nottingham University 1967 (651,15000,23)	Conflict-free, berhasil implementasi, kurang optimal
Fisher & Shier (1983) [3]	Cluster Approach with TSP	Clemson University (---,10875,18)	Diuji pada real data, tapi tidak diimplementasi
Leong & Yeong (1990) [4]	Cluster Approach with QAP	National University of Singapore (800,16000,44)	Conflict-free, diuji pada real data, tidak diimplementasi
Johnson (1990) [5]	Cluster Approach with Simulated Annealing	University of the South Pacific (200,2350,20)	Conflict-free, sukses dalam implementasi
Lotfi & Cerveny (1991) [6]	Cluster Approach with QAP and TSP	Suny al Buffalo (858,11331,15)	Minimize conflicts, diimplementasi
Hertz (1991) [7]	Tabu Search	Swiss Federal Institute of Technology in Zurich (1851,4529,---)	Minimize conflicts, diuji pada real data, tidak diimplementasi
Balakhrisnan, Luccena & Wong (1992) [8]	Cluster Approach with TSP	Purdue University (3569,---,30)	Minimize conflicts, diuji pada real data, tidak diimplementasi
Carter, Laporte & Chinneck (1994) [9]	Sequential with Backtracking	University of Toronto, Faculty of Engineering (200,2400,20) dan Carleton University (682,16925,34)	Minimize conflicts, waktu proses lama, diimplementasi
Thompson & Dowland (1995) [10]	Simulated Annealing	Swansea University 1992 (600,3000,24)	Conflict-free, berhasil diimplementasi
Carter, Laporte & Lee (1995) [11]	Sequential with Backtracking	11 institusi (81-2419,611-30032, 20-51)	Minimize conflicts, waktu proses lama, diimplementasi



Gambar 1. DF Search



Gambar 2. BF Search

- j) Kelas-kelas matakuliah yang lebih dari 1 pertemuan per minggu harus mendapat jadwal yang berurutan. Kode : JP (Jumlah Pertemuan). Proses perkuliahan untuk matakuliah yang memiliki bobot lebih dari 2 SKS (1 pertemuan) akan berjalan jauh lebih efektif apabila pertemuan-pertemuan tersebut dijadwalkan berurutan.

3.2. *Soft Constraint*

Berbeda dari *hard constraint*, kendala yang termasuk dalam kategori *soft constraint* adalah kendala yang tidak selalu dapat terpenuhi dalam proses pembentukan jadwal kuliah. Meskipun tidak harus dipenuhi, tetapi jadwal kuliah yang dihasilkan harus semaksimal mungkin berusaha memenuhi ketentuan *soft constraint* ini. Semakin tinggi persentase pemenuhan *soft constraint*, semakin tinggi pula kualitas dan tingkat kesempurnaan jadwal kuliah yang dihasilkan.

Berikut adalah daftar *soft constraint* dalam masalah penjadwalan kuliah di Universitas Bina Nusantara beserta penjelasannya :

- a) Untuk setiap kelas yang dibuka diusahakan ada dosen yang mengajar. Kode : KDSN (Ketersediaan Dosen). Meskipun idealnya semua kelas memiliki dosen pengajar, pada saat proses pembuatan jadwal berlangsung tidak semua kelas dapat memperoleh alokasi dosen. Kelas-kelas yang belum mendapat dosen akan dicari dosen oleh Ketua Jurusan setelah proses pembuatan jadwal selesai.
- b) Seorang dosen tidak boleh mendapatkan jadwal mengajar lebih dari 3 pertemuan dalam sehari. Kode : maxD (Maksimal Dosen). Berdasarkan penelitian akademis, apabila seorang dosen dijadwalkan mengajar lebih dari 3 pertemuan dalam sehari, maka proses perkuliahan di kelas ke-4 dan seterusnya dalam hari itu akan menjadi tidak efektif.
- c) Seorang dosen diusahakan tidak mendapat jadwal mengajar hanya 1 pertemuan dalam sehari. Kode minD (Minimal Dosen). Dengan mempertimbangkan faktor jarak dan efisiensi waktu, seorang dosen yang rumahnya cukup jauh akan dirugikan apabila dosen itu harus datang ke kampus untuk mengajar hanya 1 pertemuan kemudian pulang kembali. Tentu saja hal ini tidak dapat diterapkan secara mutlak karena ada sejumlah dosen yang memang hanya bersedia mengajar satu pertemuan saja pada hari tertentu.
- d) Mahasiswa paket tidak boleh mendapatkan jadwal kuliah lebih dari 3 pertemuan dalam sehari. Kode : maxP (Maksimal Paket). Seperti yang dialami dosen, apabila seorang mahasiswa harus kuliah lebih dari 4 pertemuan dalam sehari, maka pertemuan ke-4 dan seterusnya akan menjadi tidak efektif.
- e) Mahasiswa paket diusahakan tidak mendapat jadwal kuliah hanya 1 pertemuan dalam sehari. Kode : minP (Minimal Paket). Seorang mahasiswa yang rumahnya cukup jauh akan merasa lebih baik apabila perkuliahan yang akan dihadapinya lebih dari 1 pertemuan dalam sehari.
- f) Jadwal mengajar dosen dalam sehari diusahakan urut tanpa jeda. Kode : urutD (Urut Dosen). Hal ini dimaksudkan agar dosen tidak perlu menunggu hingga jam mengajar berikutnya.
- g) Jadwal kuliah mahasiswa dalam sehari diusahakan urut tanpa jeda. Kode : urutP (Urut Paket). Hal ini dimaksudkan agar mahasiswa paket tidak perlu menunggu hingga jam kuliah berikutnya.
- h) Jadwal mengajar dosen dalam sehari diusahakan di dalam satu lokasi kampus. Kode : locD (Lokasi Dosen). Ketentuan ini dibuat agar sedapat mungkin seorang dosen tidak perlu berjalan dari satu kampus ke kampus lainnya untuk mengajar dalam sehari.
- i) Jadwal kuliah mahasiswa paket dalam sehari diusahakan di dalam satu lokasi kampus. Kode : locP (Lokasi Paket).
- j) Jumlah mahasiswa yang dialokasikan ke dalam satu ruangan kelas diusahakan sedekat mungkin dengan kapasitas ruangan tersebut. Kode : ER (Efisiensi Ruang). Batasan ini dilakukan untuk menghindari sebuah kelas yang hanya terdiri dari 10 orang dijadwalkan di ruangan berkapasitas 90 sementara masih banyak kelas lain yang lebih membutuhkan kelas tersebut.
- k) Kelas-kelas matakuliah yang sama diusahakan untuk terjadwal di hari dan jam yang berbeda. Kode : PM (Penyebaran Matakuliah). Hal ini dimaksudkan untuk mempermudah pihak Jurusan dalam mengisi dosen bagi kelas-kelas yang belum memperoleh alokasi dosen.

4. BEBERAPA TEKNIK *SEARCH TREE*

Untuk mencari solusi dengan menggunakan *search tree* dapat digunakan beberapa teknik berikut [1] :

- **Depth First Search (DFS)** : Metode ini menelusuri simpul-simpul yang ada pada sebuah cabang *tree* sampai simpul terdalam / terbawah. Apabila solusi tidak ditemukan, maka pencarian dilakukan pada level atas cabang yang lain. Gambar 1. menyajikan cara kerja DFS.
- **Breadth First Search (BFS)** : Metode ini menelusuri simpul-simpul pada satu level hingga selesai dulu, baru berpindah ke level selanjutnya dalam *tree*. Gambar 2. menyajikan cara kerja BFS.
- **Heuristic Search** : Metode *heuristic* ini merupakan *Breadth First Search* dengan tambahan adanya satu

atau lebih strategi khusus untuk mencoba memilih simpul cabang yang memiliki kemungkinan lebih baik untuk mengarah pada solusi yang benar. Pada tiap *level*, program akan menganalisa semua simpul *level* di bawahnya untuk mencari kemungkinan terbaik mendapat solusi, jadi ada kemungkinan sejumlah simpul atau cabang sama sekali tidak pernah ditelusuri.

➤ **Intelligent/Smart Backtracking** : sebuah teknik yang banyak digunakan sebagai penerapan *heuristic* dalam memecahkan suatu masalah. Pada contoh gambar 1, apabila sistem menemukan jalan buntu pada simpul nomor 3, maka alur algoritma tidak harus melakukan *backtracking* ke simpul nomor 2 melainkan bisa saja langsung menuju simpul nomor 1. Lompatan yang tidak biasa ini mungkin dilakukan apabila memang terdapat sebuah perhitungan untuk menghitung simpul mana yang harus menjadi tujuan. Perhitungan ini sendiri berbeda untuk setiap kasus yang dihadapi.

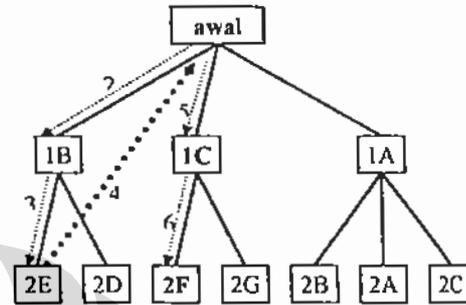
➤ **Look Ahead** : Dengan penggunaan teknik *Look Ahead*, sebuah *search tree* ditelusuri dengan kombinasi *breadth first search* dan *depth first search*. Setiap kali proses sampai pada sebuah simpul, maka proses akan mengumpulkan dan menganalisa semua simpul di bawahnya untuk menentukan simpul yang memiliki kemungkinan terbaik untuk mendapatkan solusi akhir. Pada contoh gambar 1, simpul yang diperiksa setelah simpul nomor 2 belum tentu simpul nomor 3. Apabila simpul nomor 5 dianggap memiliki kualitas lebih baik, alur dapat langsung memilih simpul nomor 5 sebagai langkah selanjutnya. Teknik *heuristic* ini banyak digunakan untuk mempersingkat *running time* secara signifikan.

Penggunaan teknik-teknik di atas sangat bervariasi dan tergantung pada kebutuhannya. Dalam beberapa kasus, sering digunakan paduan dua atau lebih teknik untuk mencapai hasil yang optimal.

5. INTI ALGORITMA

Algoritma *Intelligent Search* adalah sebuah algoritma pencarian yang merupakan kombinasi dari pencarian *Breadth First Search* dan *Depth First Search*, dipadukan dengan teknik *Smart Backtracking* dan *Look Ahead*. Secara kategori, algoritma ini merupakan perpaduan antara algoritma *Sequential Approach* dengan *Constraint-Based Approach*. Tabel 2 menyajikan modul utama dari algoritma *intelligent search*. Gambar 4 menyajikan teknik *intelligent search*. Teknik *smart backtracking* (lihat gambar 3) juga diterapkan untuk optimasi proses yang dapat meminimalkan waktu *running program*.

Meskipun memiliki struktur logika berupa *search tree*, algoritma *Intelligent Search* dengan *Smart Backtracking* memiliki struktur pemrograman yang cenderung linear dengan adanya sejumlah *flags* pengontrol proses. *Boolean flags* ini berfungsi untuk mengatur dan mengkoordinasikan modul-modul utama yang terdiri atas cari dosen, cari ruang dan waktu, pengurutan prioritas waktu, dan proses *backtracking*.



Gambar 3. Langkah Smart Backtracking

Kelas dan matakuliah yang merupakan komponen untuk dijadwalkan, ditetapkan sebagai variabel, sedangkan komponen dosen, waktu, dan ruang ditetapkan sebagai value. Kombinasi yang dapat diperoleh dari komponen-komponen ini akan menyebabkan waktu pencarian yang meningkat secara eksponensial apabila pencarian solusi tidak dilakukan secara *intelligent*.

Constraint $\max D$, $\min D$, $\max P$, dan $\min P$ menyebabkan proses pengalokasian dosen dan kelas paket pada suatu waktu cenderung memiliki banyak keterbatasan. Hal ini justru dimanfaatkan dengan membuat dasar perhitungan *Look Ahead* yang akan menjelajahi cabang pohon yang terbaik terlebih dahulu berdasarkan perhitungan ke depan berdasarkan jumlah jam mengajar/jam kuliah dosen dan kelas paket.

Smart Backtracking dilakukan dengan memperhitungkan irisan keterkaitan antara 2 kelas. Dua kelas dikatakan memiliki keterkaitan apabila :

- Matakuliah kelas A sama dengan matakuliah kelas B
- Satu atau lebih dosen yang boleh mengajar matakuliah kelas A juga dapat mengajar matakuliah kelas B
- Kelas A dan kelas B memiliki satu atau lebih sub-paket yang sama

Penggunaan kombinasi dari 4 teknik dasar di atas telah terbukti mampu menyelesaikan masalah penjadwalan kuliah dengan baik. Tidak ada *hard constraint* yang terlanggar dalam setiap jadwal kuliah yang dihasilkan. Pemuasan *soft constraint* tergantung pada karakteristik data input yang dimasukkan.

Tabel 2. Algoritma Modul Utama Metode Intelligent Search

```

N = 0
Ntercapai = 0
LakukanBacktracking = False
LakukanCariDosen = True
LakukanCariRuang = True
LakukanCariKelasMTK = True
While N <= JumKelasMTK do
  If LakukanCariKelasMTK then
    If LakukanBacktracking then
      Ambil posisi Dosen, Ruang dan Waktu
    else
      Set Dosen, Ruang, dan Waktu = 0
    end if
  end if
  If LakukanCariDosen then CariDosen(N,D)
  If LakukanCariRuang then CariRuang(N,R)
  Update data Ntercapai
  If N = JumKelasMTK then
    If IsValid() then
      SalinHasil()
      Keluar dari proses
    end if
  end if
  If TidakDapatRuang then
    Siapkan iterasi berikutnya
    If TidakDapatDosen then
      Atasi kondisi non-ideal tidak ada dosen
      Siapkan iterasi (smart backtracking)
    end if
  else
    If N < JumKelasMTK then
      Periksa KelasMTK berikutnya
      Siapkan iterasi berikutnya
    else
      Siapkan iterasi berikutnya
    end if
  end if
end while
    
```

5.1. Smart Backtracking

Apabila sebuah variabel kelas-matakuliah tidak dapat memperoleh alokasi dosen atau ruangan dan waktu, maka algoritma *search tree* biasa akan melakukan *backtracking* ke variabel kelas-matakuliah sebelumnya. Penelitian yang telah dilakukan menunjukkan bahwa proses *backtracking* konvensional ini dapat diperbaiki dengan cara mencari variabel kelas-matakuliah yang memiliki keterkaitan dengan kelas-matakuliah sekarang.

Sebuah kelas A dikatakan memiliki keterkaitan dengan kelas B apabila :

- Matakuliah kelas A sama dengan matakuliah kelas B
- Satu atau lebih dosen yang boleh mengajar matakuliah kelas A juga dapat mengajar matakuliah kelas B
- Kelas A dan kelas B memiliki satu atau sub paket yang sama

Penerapan teknik *smart backtracking* pada algoritma *Intelligent Search* memungkinkan proses *backtracking*

langsung mengarahkan *pointer current class* pada kelas yang memenuhi salah satu dari kriteria di atas. Hal ini dapat menghemat cukup banyak iterasi (tergantung dari karakteristik data *input*) yang tentu saja akan mempersingkat waktu proses. *Tabel 3.* akan menjelaskan lebih lanjut tentang teknik *smart backtracking* pada algoritma *Intelligent Search*.

Tabel 3 Algoritma Pengecekan Kelas untuk Smart Backtracking

```

Result = False
If MTK kelas A = MTK kelas B then
  Result = True
  Exit
end if
If Dosen kelas B terdapat dalam himpunan Dosen kelas A then
  Result = True
  Exit
end if
If Sub Paket kelas A dan B memiliki kesamaan then
  Result = True
  Exit
end if
    
```

6. HASIL EVALUASI

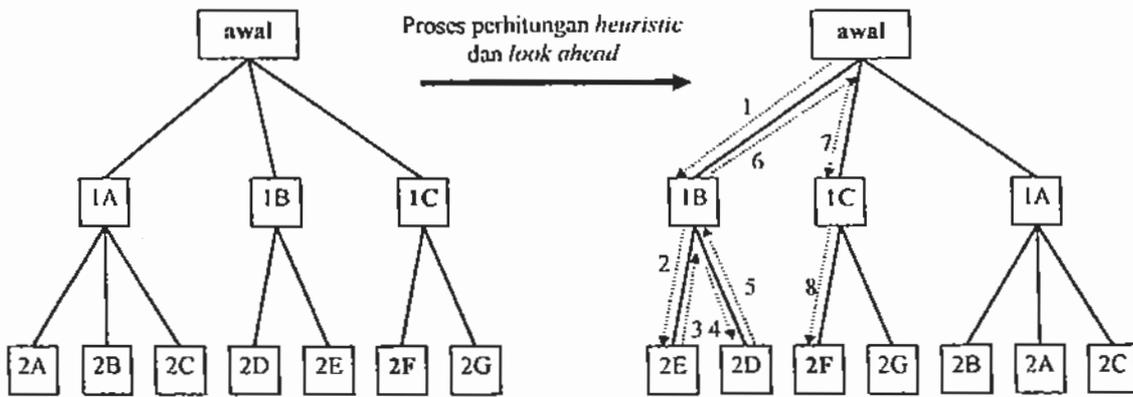
Algoritma yang kami kembangkan ini telah dicoba dengan sejumlah studi kasus berskala kecil (7 dosen, 7 matakuliah, 23 kelas, 2 ruang, 35 jam perkuliahan tiap minggu, dan lebih dari 1380 mahasiswa) dan menghasilkan output yang diinginkan dalam waktu yang sangat singkat. Percobaan dengan *real data* (1198 dosen, 1457 matakuliah, 2311 kelas, 122 ruang, 40 jam perkuliahan tiap minggu, dan lebih dari 20000 mahasiswa) telah menghasilkan solusi walaupun masih terus dilakukan proses efisiensi. Sejumlah constraint, terutama yang berkaitan dengan dosen dan mahasiswa kelas paket sering sekali sulit dipenuhi karena adanya sejumlah kelas yang merupakan gabungan dari beberapa paket (dapat mencapai 12 paket).

Pada tabel 4 dan 5, kami sajikan karakteristik dari salah satu data percobaan serta rangkuman hasil yang diperoleh.

7. KESIMPULAN

Algoritma *Intelligent Search* yang dibantu dengan *Smart Backtracking* dapat digunakan untuk menyelesaikan masalah penjadwalan kuliah di Universitas Bina Nusantara dengan lebih baik dari segi waktu dibanding proses yang sedang berjalan. Keberadaan sistem paket kuliah di universitas tersebut merupakan *constraint* yang paling berat dalam penyusunan jadwal kuliah.

Meskipun proses pencarian solusi difokuskan untuk memecahkan masalah penjadwalan kuliah sesuai dengan kasus nyata di universitas tersebut, dengan sedikit perbaikan, algoritma ini diharapkan mampu menyelesaikan masalah penjadwalan kuliah di universitas lain.



Gambar 4. Teknik Intelligent Search

Tabel 4. Karakteristik Data Percobaan

	Pembandingan	Percobaan
1	Jumlah dosen aktif	4
2	Jatah SKS dosen (maks, rata2, std. deviasi)	16 / 8.5 / 5.26
3	Jumlah matakuliah dosen (maks, rata2, std. deviasi)	2 / 1.25 / .5
4	Jumlah waktu kesediaan mengajar per minggu (maks, rata2, std. deviasi)	8 / 4.5 / 3
5	Jumlah matakuliah	4
6	Jumlah dosen untuk tiap matakuliah (maks, rata2, std. deviasi)	2 / 1.25 / .5
7	Jumlah ruangan	1
8	Rata-rata kapasitas ruangan	80
9	Jumlah jam perkuliahan per minggu	12
10	Jumlah paket	3
11	Jumlah sub paket	3
12	Jumlah kelas paket	4
13	Jumlah kelas multipaket	1
14	Persentase kelas multipaket	25
15	Jumlah sub paket dalam kelas multipaket (maks, rata2)	2 / 2
16	Jumlah record kelas paket	5
17	Jumlah record kelas multipaket	2
18	Persentase record kelas multipaket	40
19	Jumlah kelas KRS	3
20	Jumlah kelas total	7
21	Persentase kelas KRS	42.86
22	Persentase kelas paket	57.14

Tabel 5. Rangkuman Jadwal Kuliah Hasil Percobaan

	Constraint	Intelligent Search
DB	Jadwal dosen bentrok	0.00%
MB	Jadwal mahasiswa paket bentrok	0.00%
RB	Ruangan bentrok	0.00%
MT	Dosen mengajar tidak sesuai matakuliah	0.00%
WD	Dosen mengajar diluar waktu kesediaan	0.00%
SD	Dosen mengajar melebihi jatah SKS	0.00%
WR	Kelas tidak mendapat waktu dan ruang	0.00%
KR	Kapasitas ruangan tidak cukup	0.00%
RK	Kelas dengan ruangan khusus tidak terjadwal dengan benar	0.00%
JP	Matakuliah >2 SKS mendapat jadwal tidak urut	0.00%
KIDSN	Kelas belum mendapat dosen pengajar	0.00%
MaxD	Dosen mengajar lebih dari 3 pertemuan dalam sehari	0.00%
MinD	Dosen mengajar kurang dari 2 pertemuan dalam sehari	20.00%
MaxP	Mahasiswa paket kuliah lebih dari 3 pertemuan	0.00%
MinP	Mahasiswa paket kuliah kurang dari 2 pertemuan	25.00%
UrutD	Jadwal mengajar dosen dalam sehari tidak urut	0.00%
UrutP	Jadwal kuliah mahasiswa paket dalam sehari tidak urut	0.00%
LocD	Dosen mendapat jadwal mengajar tidak di satu kampus dalam sehari	0.00%
LocP	Mahasiswa paket mendapat jadwal kuliah tidak di satu kampus dalam sehari	0.00%
ER	Efisiensi penggunaan ruangan	100.00%
PM	Penyebaran hari dan jam untuk matakuliah yang sama	100.00%

REFERENSI

[1] T. H. Cormen, C. E. Leiserson, dan R. L. Rivest, *Introduction to Algorithms*. Massachusetts Institute of Technology, USA, The MIT Press, 1990.

[2] E. Foxley dan K. Lockyer, The Construction of Examination Timetables by Computer. *The Computer Journal*, vol. 11, 1968, 264-268.

[3] J. G. Fisher dan D. R. Shier, A Heuristic Procedure for Large-Scale Examination Scheduling Problems. *Technical Report 417, Dept of Mathematical Sciences, Clemson University*, 1983.

[4] T. Y. Leong dan W. Y. Yeong, *A Hierarchical Decision Support System for University Examination Scheduling*. National University of Singapore, 1990.

[5] D. Johnson, Timetabling University Examinations. *Journal of The Operational Research Society*, vol 41, 1990, 39-47.

[6] V. Lotvi dan R. Cervený, A Final-exam-scheduling Package. *Journal of the Operational Research Society*, vol 42, 1991, 205-216.

[7] A. Hertz, Tabu Search for Large Scale Timetabling Problems. *European Journal of Operations Research*, vol 54, 1991, 39-47.

[8] N. Balakrishnan, A. Lucena, dan R. T. Wong, Scheduling Examinations to Reduce Second-order Conflicts. *Computers & Operations Research*, vol 19, 1992, 353-361.

[9] M. W. Carter, G. Laporte, dan J. W. Chinneck, A General Examination Scheduling System. *Interfaces* 24, 1994, 109-120.

[10] J. M. Thompson dan K. A. Dowsland, *Variants of Simulated Annealing for the Examination Timetabling Problem*. European Business Management School, Swansea University, Swansea, United Kingdom. 1995.

[11] M. W. Carter, G. Laporte, dan S. Y. Lee, Examination Timetabling: Algorithmic Strategies and Applications. *Journal of the Operational Research Society*.1995