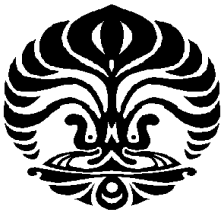
PENJADWALAN KULIAH DENGAN ALGORITMA MEMETIKA

LISMANTO

0304017042



UNIVERSITAS INDONESIA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

DEPARTEMEN MATEMATIKA

DEPOK

2008

PENJADWALAN KULIAH DENGAN ALGORITMA MEMETIKA

Skripsi diajukan sebagai salah satu syarat

untuk memperoleh gelar Sarjana Sains

Oleh :

LISMANTO

0304017042

DEPOK

2008

SKRIPSI : PENJADWALAN KULIAH DENGAN MENGGUNAKAN ALGORITMA MEMETIKA

NAMA : LISMANTO

NPM : 0304017042

SKRIPSI INI TELAH DIPERIKSA DAN DISETUJUI

DEPOK, 15 uli 2008

Dr. Zuherman Rustam, DEA	**Dr. Yudi Satria, M.T**
PEMBIMBING I	PEMBIMBING II

Tanggal lulus Ujian Sidang Sarjana : 15Juli 2008

Penguji I : Dr. Zuherman Rustam, DEA

Penguji II : Dra. Denny Riama Silaban, M.Kom

Penguji III : Mila Novita, S.Si, M.Si

KATA PENGANTAR

Sebelumnya tak lupa kita panjatkan puji syukur kepada Tuhan Yang Maha Esa, sehingga dengan hidayah-Nya penulis dapat menulis skripsi ini. Terima kasih sepenuhnya penulis ucapkan untuk Pak Zuherman dan Pak Yudi Satria yang telah bersama-sama membimbing skripsi ini. Terima kasih kepada keluargaku yang telah memberi doa untuk kelancaran penulisan skripsi ini. Terima kasih juga kepada teman-teman Matematika UI yang telah memberi dorongan dan dukungan dalam penulisan skripsi ini.

Dalam rangka memenuhi tugas akhir program pendidikan Sarjana Matematika UI, maka penulis menyusun sebuah skripsi dengan judul "Penjadwalan Kuliah dengan Algoritma Memetika". Skripsi ini dimaksudkan untuk mengeksplorasi konsep penjadwalan kuliah secara umum dan secara khusus di departemen Matematika UI. Skripsi ini ditulis sesuai dengan kondisi terakhir penjadwalan kuliah di departemen Matematika UI, ditambah paparan tentang pembahasan dan analisis proses pencarian jadwal kuliah yang bagus.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan. Adalah menjadi harapan penulis apabila terdapat kritik dan saran yang membangun sehingga penulis dapat mengadakan perbaikan untuk masa yang akan datang.

Penjadwalan kuliah..., Lismanto, FMIPA UI, 2008

Sebagai penutup, penulis mengucapkan terima kasih kepada semua pihak yang telah membantu dalam penyusunan skripsi ini. Semoga skripsi ini dapat berguna bagi diri penulis serta pembacanya.

Jakarta, 17 Juli 2008

Penulis

Penjadwalan kuliah..., Lismanto, FMIPA UI, 2008

ABSTRAK

Masalah penjadwalan kuliah adalah masalah optimasi yang komputasinya rumit karena terdapat sejumlah ruangan dengan kapasitas tertentu, sejumlah dosen, serta sejumlah mahasiswa yang akan mendefinisikan kendala *hard* dan *soft* (Salwani, 2007). Penjadwalan kuliah pernah dilakukan dengan *Simulated anneling* (Elfitriadi, 2001), *tabu search* (Herlina, 2000) dan *iterated local search* (Lourenco, Martin dan Stutzle, 2002). *Simulated anneling* kurang efektif dalam pencarian solusi kendala *hard*, algoritma genetika tidak menjamin solusi optimal global, sedangkan *iterated local search* kurang efektif dalam optimasi kendala *soft*. Dalam skripsi ini, pembuatan jadwal dilakukan dengan menggabungkan algoritma genetika dan *iterated local search* disebut dengan algoritma memetika. Penambahan *iterated local seacrh* inilah yang memungkinkan dalam pencarian jadwal terbaik (optimal global). Data yang digunakan diperoleh dari departemen Matematika UI semester genap tahun 2008 dan hasilnya yaitu seluruh kendala *hard* cepat terpenuhi dan mencapai solusi optimal global dengan waktu komputasi pada komputer dual core 3.0GHz, 2GB RAM yang kurang dari 2 menit.

Kata kunci : genetika, *local search*, memetika, penjadwalan kuliah

vii + 42 hlm; lamp

Bibliografi : 8 (2001-2007)

DAFTAR ISI

Penjadwalan kuliah..., Lismanto, FMIPA UI, 2008

Penjadwalan kuliah..., Lismanto, FMIPA UI, 2008

DAFTAR GAMBAR

Penjadwalan kuliah..., Lismanto, FMIPA UI, 2008

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Masalah penjadwalan secara umum adalah aktifitas penugasan yang berhubungan dengan sejumlah kendala, sejumlah kejadian yang dapat terjadi pada suatu periode waktu dan tempat / lokasi sehingga fungsi objektif sedekat mungkin dapat terpenuhi. Masalah ini muncul di berbagai bidang kegiatan maupun instansi seperti rumah sakit, universitas, penerbangan, pabrik dan lain-lain. Desain model masalah itu bervariasi sesuai dengan kebutuhan serta keadaan kendala di lapangan.

Pada penjadwalan kuliah sejumlah kuliah harus dijadwalkan ke dalam ruang dan slot-waktu tertentu, dimana sejumlah kuliah tidak boleh bentrok, dosen hanya dapat mengajar pada waktu kehadiran, mahasiswa dapat memperoleh kuliah wajib atau pilihan tanpa bentrokan, serta batasan-batasan yang disesuaikan dengan kondisi di lapangan. Oleh karena itu kombinasi dari jadwal kuliah menjadi sangat beragam dan rumit sehingga tidak mudah untuk mengukur dan mencari jadwal yang paling baik.

Pengukuran kualitas jadwal dilakukan dengan pengelompokan dua kendala yaitu *hard* dan *soft*, dimana jadwal yang layak harus memenuhi

seluruh kendala *hard* dan kualitas jadwal ditentukan oleh penalti kendala *soft* yang mengindikasikan banyaknya pelanggaran jadwal terhadap kendala *soft*. Kendala *hard* dan *soft* didefinisikan oleh pengguna, atau ditentukan sesuai dengan kondisi penjadwalan yang ada, seperti kondisi penjadwalan kuliah di departemen Matematika UI yang membagi mahasiswa menjadi 7 kelompok yakni kelompok tingkat1, tingkat 2, serta 5 bidang minat.

Penjadwalan kuliah merupakan masalah optimasi yang komputasinya rumit karena terdapat sejumlah ruangan dengan kapasitas tertentu, sejumlah dosen, serta sejumlah mahasiswa yang akan mendefinisikan kendala *hard* dan *soft* (Salwani, 2007).Telah banyak cara yang dilakukan untuk mencari jadwal yang baik seperti dengan menerapkan algoritma genetika, *Simulated anneling* (Elfitriadi, 2001), *tabu search* dan *iterated local search*. *Simulated anneling* dan *iterated local search* kurang efektif dalam pencarian jadwal yang bagus (Rossidoria, 2006), begitu juga dengan metode *tabu search* kurang efektif (Herlina, 2000).

Algoritma genetika dinilai efektif tetapi seperti telah diketahui bahwa algoritma tersebut tidak menjamin diperoleh jadwal yang paling baik (jadwal yang optimal). Dalam paper ini pembuatan jadwal dilakukan dengan menggabungkan algoritma genetika dan *iterated local search* yang disebut dengan algoritma memetika. Diharapkan dengan penambahan *iterated local seacrh* dapat menambah kemungkinan dalam mencari jadwal kuliah yang paling baik (jadwal yang optimal global).

1.2 PERUMUSAN MASALAH

Sejumlah kuliah akan dijadwalkan pada slot-waktu dan ruangan dengan batasan atau kendala yang akan ditentukan, bagaimana membuat jadwal kuliah dengan algoritma memetika sehingga jadwal kuliah dengan kualitas maksimal?

1.3 TUJUAN

1. Membuat jadwal kuliah yang layak, yakni:
   - Memenuhi kendala *hard*
   - Memenuhi sebanyak mungkin kendala *soft*
2. Melihat kinerja algoritma memetika untuk penjadwalan kuliah

1.4 BATASAN MASALAH

Batasan masalah dalam skripsi ini adalah sebagai berikut:

1. Disesuaikan dengan kondisi penjadwalan di dept. Matematika UI
2. Mahasiswa terbagi menjadi 7 kelompok yakni kelompok tingkat1, tingkat 2, serta 5 bidang minat

1.5 SISTEMATIKA PENULISAN

Skripsi ini dibagi dalam lima bab yakni bab I berisi pendahuluan, bab II berupa landasan toeri yang membahas konsep dasar masalah penjadwalan kuliah, algoritma memetika serta komponen algoritma memetika. Bab III berisi penyelesaian masalah penjadwalan kuliah di departemen matematika UI dengan algoritma memetika. Selanjutnya dalam Bab IV berisi analisis hasil percobaan dan bab V berisi kesimpulan.

BAB II

LANDASAN TEORI

Pada bab ini akan dibahas mengenai konsep dasar masalah penjadwalan kuliah, algoritma memetika serta komponen algoritma memetika. Agoritma memetika diilhami dari proses evolusi makhluk hidup, sedangkan penjadwalan kuliah merupakan bagian dari penjadwalan dengan kriteria dan kendala tertentu. Hubungan dari kedua hal tersebut adalah algoritma memetika dapat digunakan untuk menyelesaikan masalah penjadwalan kuliah.

2.1 PENJADWALAN KULIAH STANDAR INTERNASIONAL

Masalah penjadwalan kuliah bermula dari kompetisi membuat jadwal kuliah di 20 instansi di Universitas Napier di Edinburgh Skotlandia. Model permasalahan telah ditetapkan, tetapi partisipan dapat menggunakan berbagai metode untuk menyelesaikannya. Berikut akan dibahas model masalah penjadwalan kuliah standar internasional dan mesin penjadwalan kuliah.

2.1.1 Deskripsi Masalah Penjadwalan Kuliah Standar Internasional

Masalah penjadwalan kuliah standar internasional dikenal dengan *University Course Timetabling Problem* (UCTP) akan menjadwalkan sejumlah kuliah dengan 45 slot-waktu yakni 9 slot-waktu per hari dari senin sampai dengan jumat. Komponen yang mempengaruhi jadwal terdiri dari kapasitas ruangan yang tersedia, waktu kehadiran dosen, mahasiswa pengikut kuliah, serta batasan-batasan yang ditentukan.

Batasan atau kendala yang terdapat dalam UCTP terbagi dalam dua kategori yaitu kendala *hard* dan kendala *soft*. Kendala *hard* merupakan kendala essensial yang harus terpenuhi dalam pembuatan jadwal, sedangkan kendala *soft* merupakan kendala nonessensial yang menentukan kualitas dari jadwal yang dihasilkan.

Kendala *hard* dan *soft* terjadi akibat interaksi antar komponen jadwal, seperti kapasitas ruang dan jumlah mahasiswa pengikut kuliah, selain itu interaksi juga dapat terjadi antar kuliah-kuliah yang tidak boleh dijadwalkan pada waktu yang bersamaan. Klasifikasi kedua kendala tersebut telah ditentukan oleh Universitas Napier karena disesuaikan dengan kondisi penjadwalan kuliah di universitas tersebut.

Berikut adalah kendala *hard* dalam UCTP, untuk membentuk solusi jadwal kuliah yang layak kendala-kendala ini harus terpenuhi:

1. H1: Dua atau lebih kuliah tidak dapat diberikan pada suatu waktu dengan ruangan yang sama
2. H2: Setiap kuliah harus dilakukan pada ruangan yang memenuhi fasilitas dan kapasitas untuk kuliah tersebut.
3. H3: Tidak ada mahasiswa yang mendapat 2 / lebih kuliah pada waktu yang sama.

Setelah seluruh kendala *hard* terpenuhi dalam pembentukan jadwal, maka akan dilanjutkan dengan meminimumkan fungsi penalti kendala *soft* yakni pelanggaran jadwal terhadap kendala *soft* sedapat mungkin diminimalisasi. Berikut adalah kendala soft yang telah didefinisikan dalam UCTP:

1. S1: Mahasiswa tidak memperoleh kuliah pada akhir slot-waktu terakhir setiap hari
2. S2: Mahasiswa tidak mendapat kuliah lebih dari 6 jam berturut- turut
3. S3: Mahasiswa tidak memperoleh hanya satu kuliah dalam 1 hari

2.1.2 Mesin Penjadwalan Kuliah

Sejarah mesin penjadwalan kuliah bermula dari kompetisi menyelesaikan masalah penjadwalan kuliah standar internasional. Pengujian dan penilaian kompetisi ini menggunakan data 20 instansi di Universitas Napier di Edinburgh Skotlandia dimana penalti *soft* terbaik dan terburuk

setiap mata kuliah telah diketahui. Kompetisi tersebut mengundang perhatian partisipan dari seluruh penjuru dunia dengan berbagai metode pendekatan.

Penilain untuk masing-masing partisipan berdasar atas penalti kendala soft yang dihitung sebagai berikut :

$$p^j = \sum_{i \in I} \frac{f_i(j_i^j) - f_i(j_i^b)}{f_i(j_i^w) - f_i(j_i^b)}$$

dimana :

j := Partisipan ke-j

i := Himpunan 20 instansi

f_i := Penalti *soft* instansi ke-i

j_i^j := Jadwal yang disusun oleh partisipan j untuk instansi ke-i

j_i^b := Jadwal terbaik pada instansi ke-i yang telah diketahui

j_i^w := Jadwal terburuk pada instansi ke-i yang telah diketahui

Penjumlahan suku-suku tersebut menghasilkan penalti untuk partisipan ke-j, sehingga pemenang kompetisi adalah partisipan dengan penalti minimum. Ternyata dengan sistem penilaian kompetisi diatas, algoritma hibrid metaheuristik menjadi algoritma yang paling efektif. Algoritma memetika juga telah diujikan dalam masalah tersebut dan hasilnya adalah seluruh kendala *hard* dan *soft* seluruh instansi terpenuhi dengan waktu komputasi yang efektif (Rossidoria, 2004). Hal itu menjadi alasan penulis untuk menggunakan algoritma memetika dalam skripsi ini.

2.2.3 Penjadwalan Kuliah di Departemen Matematika UI

Penjadwalan kuliah di departemen Matematika UI melibatkan beberapa komponen yakni sejumlah ruang kuliah, dosen, serta mahasiswa. Penjadwalan kuliah awal tahun 2008 tersedia 8 ruang kuliah dengan beragam kapasitas, 35 dosen yang memberikan kuliah pada jam kehadiran, serta mahasiswa terbagi atas 7 kelompok yaitu kelompok mahasiswa tingkat 1, tingkat 2, serta 5 bidang minat.

Kendala dalam pembentukan model penjadwalan kuliah di departemen matematika UI dibuat melalui pengambangan UCTP serta penyesuaian dengan kondisi penjadwalan perkuliahan di departemen Matematika UI. Pengembangan dan penyesuaian itu mengakibatkan penambahan 2 kendala *hard* serta 1 kendala *soft* yang didefinisikan sebagai berikut:

Kendala *hard*:

1. H1: Dua atau lebih kuliah tidak dapat diberikan pada suatu waktu dengan ruangan yang sama
2. H2: Setiap kuliah harus dilakukan pada ruangan yang memenuhi fasilitas dan kapasitas untuk kuliah tersebut.
3. H3: Tidak ada mahasiswa yang mendapat 2 / lebih kuliah pada waktu yang sama.

4. H4: Dosen ditugaskan hanya pada waktu kehadiran
5. H5: Dosen tidak dapat memberikan 2 atau lebih kuliah pada waktu yang bersamaan

Kendala *soft*:

1. S1: Mahasiswa tidak memperoleh kuliah pada akhir slot-waktu terakhir setiap hari
2. S2: Mahasiswa tidak mendapat kuliah lebih dari 6 jam berturut- turut
3. S3: Mahasiswa tidak memperoleh hanya satu kuliah dalam 1 hari
4. S4: Kuliah tidak dilangsungkan bersamaan dengan prasyaratnya.

Penambahan kendala tersebut merupakan penyesuaian dengan kondisi penjadwalan kuliah di departemen Matematika UI yang akan diselesaikan menggunakan algoritma memetika.

2.2 KONSEP DASAR ALGORITMA MEMETIKA

Pengertian Memetika adalah suatu ilmu pengetahuan yang mempelajari proses replikasi (*replication*), penyebaran (*spread*), dan evolusi (*evolution*) dari *meme* atau pola informasi. *Meme* dapat direplikasi melalui komunikasi antar sesama individu, tanpa ada keterkaitan dengan tingkat replikasi dari gen masing-masing individu dan dapat disebarkan ke individu

lain melalui berbagai cara seperti pencarian lokal, pindah silang, evolusi atau dengan cara lainnya.

Proses replikasi *meme* disebut juga *process of self-reproduction* atau *the memetic life-cycle* yang dilanjutkan dengan penyebaran *meme.* Penyebaran terjadi melalui sebuah kelompok dari beberapa individu. Urutan proses tersebut mengilustrasikan *meme* sebagai suatu *replicator* seperti layaknya suatu gen.

Proses replikasi pada gen hanya dapat terjadi dari orang tua kepada anaknya (*vertical transmission*) yang merupakan konsep dari algoritma genetika, sedangkan proses replikasi pada *meme* dapat juga terjadi dalam satu individu (*horizontal transmission*). Gabungan kedua konsep itu digunakan sebagai dasar algoritma memetika, selanjutnya *meme* dalam algoritma memetika juga disebut dengan gen.

Individu dalam algoritma genetika dan memetika direpresentasikan dengan satu kromosom, kromosom inilah yang mewakili kandidat solusi permasalahan. Kromosom terdiri atas beberapa gen yang membawa informasi bagian solusi permasalahan, sedangkan kumpulan dari kromosom biasa disebut dengan populasi.

Algoritma memetika adalah pendekatan yang dilakukan berdasarkan populasi kromosom untuk pemecahan suatu masalah optimasi. Karena algoritma memetika hanya suatu pendekatan maka algoritma tersebut termasuk dalam metode heuristik yakni suatu metode yang digunakan ketika ukuran ruang pencarian solusi sulit dikontrol secara eksak dan belum ada

algoritma yang dapat mencari solusi optimal secara efektif. Algoritma memetika memiliki kompleksitas yang lebih tinggi dibandingkan algoritma genetika (untuk domain tertentu) karena adanya penambahan pencarian solusi lokal.

2.3 KOMPONEN ALGORITMA MEMETIKA

Pada dasarnya algoritma memetika terdiri dari tujuh komponen yaitu skema pengkodean, fungsi penalti, seleksi orang tua, pindah silang (*crossover*), pencarian lokal, mutasi, dan pergantian populasi. Berikut akan dijelaskan mengenai masing- masing komponen di atas.

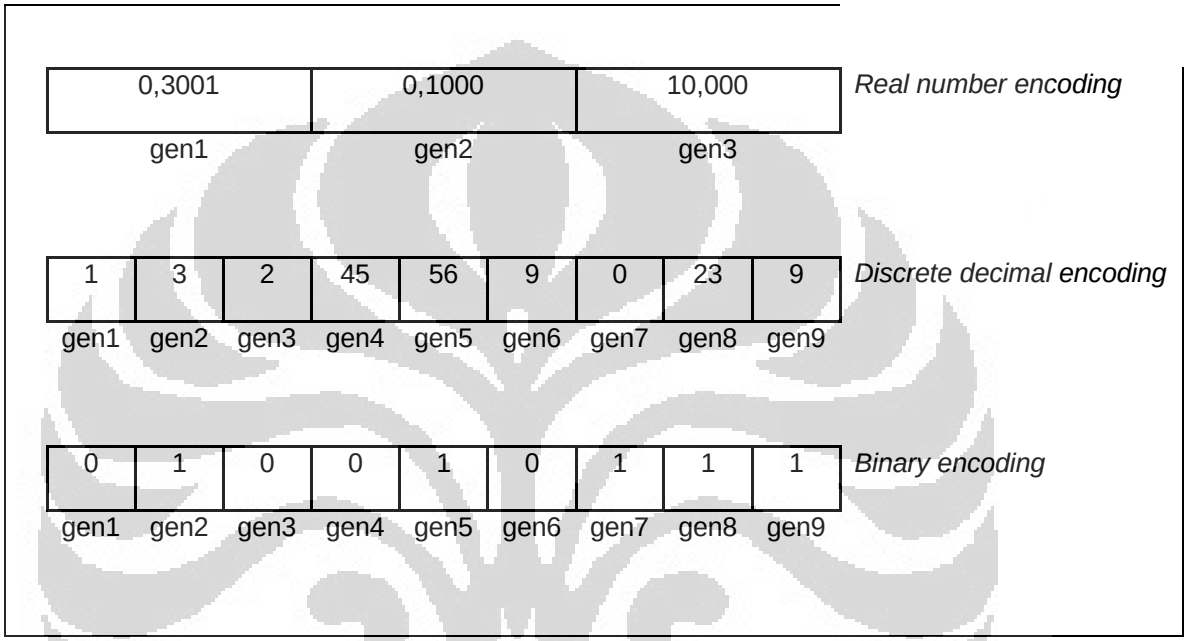
2.3.1 Skema pengkodean

Terdapat 3 skema yang paling umum digunakan yakni :

- *Real number encoding* yaitu pengkodean nilai gen berada dalam interval [0, R], dimana R adalah bilangan positif yang biasanya bernilai 1
- *Discrete decimal encoding* yaitu pengkodean nilai gen bisa bernilai dari salah satu bilangan bulat dalam interval [0, N]

- *Binary encoding* yakni setiap gen hanya bisa bernilai 0 atau 1

Berikut merupakan gambaran skema pengkodean gen-gen dalam kromosom :



Gambar 1: Pengkodean kromosom dalam algoritma memetika

2.3.2 Fungsi penalti

Fungsi penalti merupakan nilai yang terdapat dalam suatu kromosom yang menentukan kualitas atau kelayakan suatu solusi masalah. Makna dari nilai menandakan penalti atas pelanggaran terhadap suatu kendala. Akibatnya, semakin kecil nilai penalti akan meningkatkan kualitas solusi, sedangkan kelayakan solusi ditentukan oleh batasan suatu nilai penalti.

Dalam masalah optimasi, nilai penalti tersebut biasanya diminimumkan sehingga diperoleh solusi dengan jumlah pelanggaran kendala yang paling sedikit.

2.3.3 Seleksi orang tua

Seleksi orang tua merupakan cara pemilihan dua kromosom sebagai orang tua. Dalam algoritma memetika, seleksi orang tua merupakan bagian dari evolusi yakni proses mendekati kandidat solusi permasalahan yang diharapkan. Sebelum proses seleksi orang tua harus terdapat populasi awal yang berisi sejumlah kromosom yang diinisialisasi terlebih dahulu baik dengan random, nonrandom atau gabungannya.

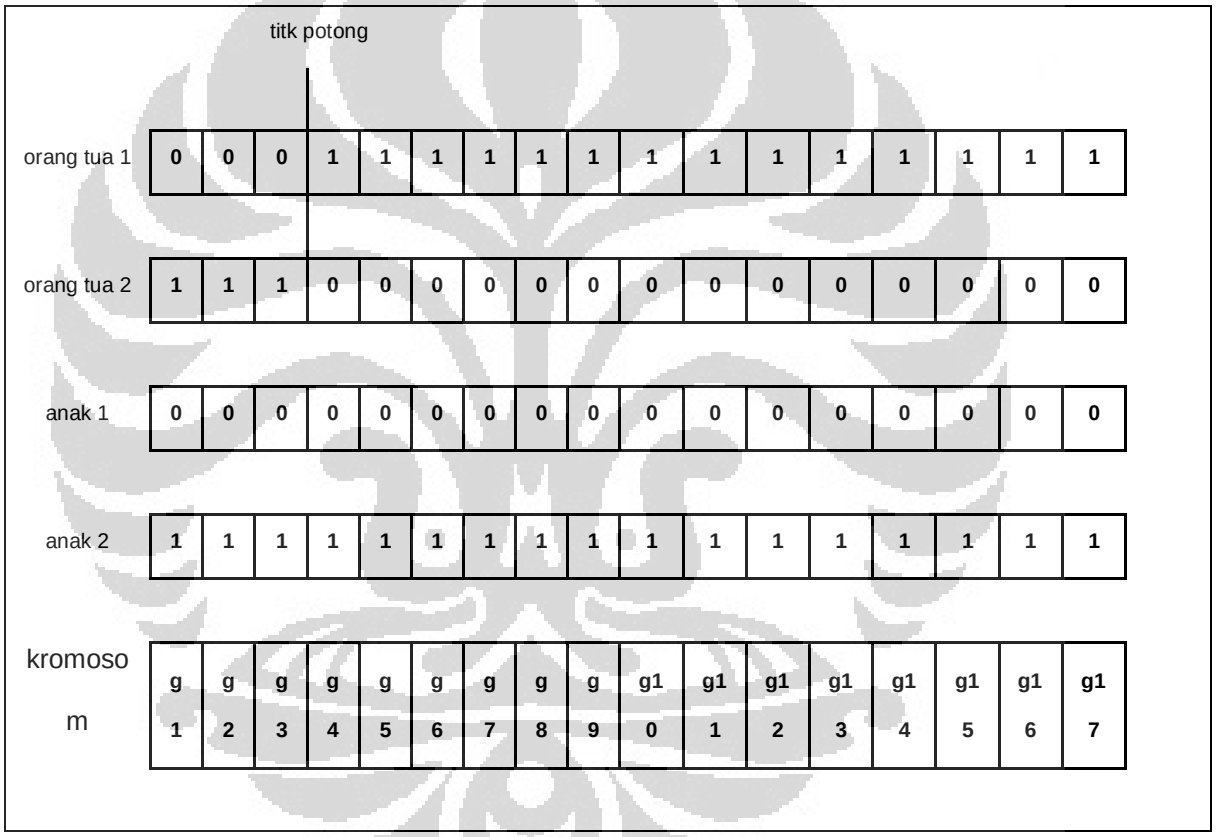
Proses seleksi dapat dilakukan dengan turnamen (*tournament selection*) atau *roulette while.* Turnamen berukuran n dilakukan dengan memilih n kromosom secara random kemudian kromosom dengan nilai penalti terbaik akan menjadi salah satu orang tua terpilih. Seleksi dengan *Roulette while* dilakukan dengan memberi proporsi yang proporsional dengan kebalikan nilai penalti setiap kromosom, kemudian salah satu orang tua dipilih secara random dengan probabilitas sebesar proporsi tersebut.

2.3.4 Pindah silang (*crossover*)

Pindah silang (*crossover*) merupakan komponen paling penting dalam algoritma genetika dan memetika karena proses inilah yang dapat mendekati kromosom-kromosom sebagai kandidat solusi yang diharapkan. Kromosom

yang mengarah pada nilai penalti yang lebih bagus diperoleh dari proses memindah-silangkan dua buah kromosom yang diperoleh dari seleksi orang tua. Pindah silang dapat menghasilkan satu atau beberapa kromosom yang biasa disebut dengan kromosom anak.

Perhatikan contoh pindah silang berikut :



Gambar 2: Pindah silang 1 titik potong

Pindah silang dapat berakibat buruk jika jumlah populasinya sangat kecil karena suatu kromosom dengan gen-gen yang mengarah ke solusi (kromosom yang diharapkan) akan sangat cepat menyebar ke kromosom-

kromosom lainnya. Untuk mengatasi masalah ini pindah silang hanya bisa dilakukan untuk probabilitas tertentu katakanlah p_c, artinya pindah silang dilakukan jika suatu bilangan random [0,1) yang dibangkitkan kurang dari p_c yang ditentukan. Pada umumnya, p_c diset mendekati satu, misalnya 0.8.

Pindah silang juga dapat dilakukan dengan cara yang berbeda-beda, yang paling sederhana adalah pindah silang satu titik potong (*one-point crossover*) seperti pada Gambar 2. Titik potong dipilih secara random, kemudian bagian pertama dari orang tua 1 digabungkan dengan bagian kedua orang tua 2 dan sebaliknya.

Untuk kasus kromosom yang sangat panjang, misalnya 2000 gen, mungkin diperlukan beberapa titik potong. Namun, penentuan jumlah titik potong ini tidak mutlak, artinya boleh berapa saja. Pindah silang dengan n titik ini sering disebut dengan *n-point crossover*, dimana n titik potong dipilih secara random dan bagian-bagian kromosom dipilih dengan probabilitas 0,5 dari salah satu orang tuanya. Salah satu skema pindah silang yang lain adalah pindah silang seragam (*uniform crossover*), yakni jika *n* titik potong dipilih sebanyak jumlah gen dikurangi satu.

2.3.5 Mutasi

Mutasi merupakan proses pertukaran / perubahan informasi yang dibawa oleh masing-masing gen dalam kromosom anak hasil pindah silang. Mutasi ini bertujuan untuk mengurangi kesamaan informasi yang dibawa oleh kromosom anak dan orang tua, sehingga dapat berkontribusi untuk menuju

kromosom yang paling berkualitas. Mutasi dilakukan pada beberapa gen dalam kromosom anak dengan probabilitas mutasi yang relatif kecil yakni $p_{mut,}$ dimana p_{mut} biasanya diset sebesar 1/n. Mutasi *adaptive* merupakan mutasi yang dilakukan pada beberapa gen sehingga tersedia tempat yang layak untuk gen-gen yang hilang akibat pindah silang.

2.3.6 Pencarian lokal

Pencarian lokal merupakan pertukaran / penempatan kembali informasi yang dibawa oleh gen-gen dalam satu kromosom anak dengan harapan dapat meningkatkan kualitas kromosom tersebut. Pencarian lokal dapat dilakukan dengan menukar dua gen, atau permutasi beberapa gen tanpa mengurangi kualitas dari kromosom sebelumya. Dalam kasus tertentu pencarian lokal hanya dilakukan kepada gen yang terkena penalti, artinya hanya informasi yang kurang tepat yang dipertukarkan antar gen dalam satu kromosom yang sama.

2.3.7 Pergantian populasi

Pergantian populasi merupakan pembentukan populasi baru yang beranggotakan kromosom-kromosom yang lebih berkualitas. Dengan pergantian populasi yang dilakukan terus menerus diharapkan kandidat solusi permasalahan semakin medekati optimal. Secara umum pergantian populasi ini dilakukan dengan memasukkan satu atau dua kromosom anak ke

populasi awal dan membuang satu atau dua kromosom dalam populasi awal yang mempunyai nilai penalti terbesar.

BAB III

PENJADWALAN KULIAH DI DEPARTEMEN MATEMATIKA DENGAN ALGORITMA MEMETIKA

Penjadwalan kuliah di departemen Matematika UI melibatkan beberapa komponen yakni ruang kuliah, dosen serta mahasiswa. Seorang dosen dapat mengajar lebih dari satu kuliah dan satu kuliah dapat diajar oleh satu atau dua dosen, tetapi seorang dosen tidak dapat memberikan dua atau lebih kuliah pada waktu yang bersamaan. Tersedia 8 ruang kuliah dengan kapasitas beragam untuk menampung sejumlah mahasiswa yang terbagi atas 7 kelompok yakni mahasiswa tingkat 1, tingkat 2 serta 5 bidang minat.

Sebelumnya akan dijelaskan terlebih dahulu mengenai komponen-komponen algoritma memetika yang berkaitan dengan penjadwalan kuliah di departemen matematika UI. Skema pengkodean yang digunakan dalam skripsi ini adalah *discret decimal encoding*, proses seleksi orang tua dilakukan dengan turnamen ukuran 2, proses pindah silang dengan *uniform crossover* yang menghasilkan 1 kromosom anak, serta mutasi hanya dilakukan untuk kuliah-kuliah yang tidak terkena penalti.

3.1 MODEL PENJADWALAN KULIAH DI DEPARTEMEN MATEMTAKA UI

Kendala *hard* dan *soft* yang didefinisikan di Bab II diperoleh dengan asumsi model yang disesuaikan dengan kondisi penjadwalan kuliah di

departemen Matematika UI . Berikut adalah asumsi yang digunakan untuk membentuk model penjadwalan kuliah di departemen Matematika UI:

1. Kelompok mahasiswa terbagi atas mahasiswa tingkat 1, tingkat 2 dan 5 bidang minat
2. Para dosen hanya dapat memberikan kuliah pada waktu tertentu.
3. Mata kuliah dengan 3 / 4 sks ditransformasi menjadi 2 kuliah

Berikut didefinisikan sejumlah komponen yang digunakan dalam formulasi model :

J := Himpunan semua jadwal yang mungkin

JL := Himpunan jadwal yang layak

H := { H1, H2, H3, H4, H5 }, yakni himpunan kendala *hard*

S := { S1, S2, S3, S4 }, yakni himpunan kendala *soft*

M := Himpunan grup mahasiswa

K := Himpunan kuliah yang akan dijadwalkan

Fungsi penalti untuk kedua kendala didefinisikan sebagai berikut :

- Kendala *hard*

$\forall j \in J$ dan $k \in K$

$$g(k) = g_1(k) + g_2(k) + g_3(k) + g_4(k) + g_5(k)$$
$$g(j) = \sum_{k \in K} g(k)$$

$g_i(k) = 0,$ jika kuliah *k* tidak melanggar kendala *hard* ke-i

$g_i(k) = 1,$ jika kuliah *k* melanggar kendala *hard* ke-i

$i = 1,2,3,4,5$

dimana *g(k)* adalah penalti kendala *hard* untuk kuliah ke-*k* dan *g(j)* adalah penalti kendala *hard* untuk jadwal *j*. Jadwal *j* disebut layak terjadi ketika *g(j)* bernilai 0.

- Kendala *soft*

$\forall j \in JL$ dan $k \in K$

$$f(k) = f_1(k) + f_2(k) + f_3(k) + f_4(k)$$
$$f(j) = \sum_{k \in K} f(k)$$

$f_i(k) = 0,$ jika kuliah *k* tidak melanggar kendala *soft* ke-i

$f_i(k) = 1,$ jika kuliah *k* melanggar kendala *soft* ke-i

$i = 1,2,3,4$

dimana *f(k)* adalah penalti kendala *soft* untuk kuliah ke-*k* dan *f(j)* adalah penalti kendala *soft* untuk jadwal *j*. Perhitungan penalti kendala *soft* dilakukan setelah jadwal layak terbentuk, dan jadwal yang optimal merupakan jadwal yang mempunyai penalti kendala *soft* terkecil.

- Fungsi tujuan $f = \min_{j \in JL} f(j)$

3.2 PENYELESAIAN MASALAH

Penyelesaian masalah penjadwalan kuliah di departemen Matematika UI dilakukan dengan bantuan software matlab 7.1. Berikut adalah prosedur

penyelesaian yang terdiri dari beberapa tahapan yaitu managemen data, representasi jadwal, pembuatan populasi awal dan pencarian solusi lokal, solusi kendala *hard* dan optimasi kendala *soft*.

3.2.1 Managemen data

Data mentah dari masalah penjadwalan kuliah merupakan interaksi antar komponen masalah yang berupa dosen, kuliah, mahasiswa serta sejumlah ruangan. Hubungan dapat terjadi antara dosen dengan kuliah yang diajarkan, mahasiswa mengambil beberapa kuliah, ruangan harus dapat menampung sejumlah peserta kuliah, beberapa kuliah tidak boleh bentrok dan sebagainya. Oleh karena itu perlu managemen data untuk mengolah data mentah ke dalam implementasi program. Berikut adalah managemen data yang digunakan :

1. Vektor kapasitas ruangan yang mengindikasikan kapasitas dari tiap-tiap ruangan yang akan digunakan untuk melangsungkan kuliah.
2. Matrik $KM_{|K|x|M|}$ yang mengindikasikan kuliah yang akan diambil oleh grup mahasiswa serta jumlah mahasiswanya, dimana |K| dan |M| masing-masing jumlah kuliah dan jumlah grup mahasiswa.

   $KM(i, j) = 0,$ jika kelompok mahasiswa ke-*j* tidak mengambil kuliah ke-i

   $KM(i, j) = n,$ jika kelompok mahasiswa ke-*j* mengambil kuliah ke-i, dan *n* adalah jumlah kelompok mahasiswa ke-*j*.

3. Matrik $WD_{|W|x|D|}$ yakni matrik yang mengindikasikan kehadiran dosen di departemen, dimana |W| dan|D| maing-masing merupakan jumlah slot-waktu dan jumlah dosen.

   $WD(i, j) = 0,$ jika dosen ke-*j* tidak dapat hadir pada slot-waktu ke-i

   $WD(i, j) = 1,$ jika dosen ke-*j* dapat hadir pada slot-waktu ke-i

4. Matrik $KD_{|K|x|D|}$, yakni matrik yang mengindikasikan dosen pengajar dari masing-masing kuliah, dengan |D| merupakan jumlah dosen yang akan ditugaskan.

   $KD(i, j) = 0,$ jika dosen ke-*j* tidak mengajar kuliah ke-*i*

   $KD(i, j) = 1,$ jika dosen ke-*j* mengajar kuliah ke-*i*

5. Daftar sks dan kuliah prasyarat dari masing-masing kuliah. Kedua daftar tersebut dapat dibuat berupa dua vektor, atau dapat digabung menjadi matrik.

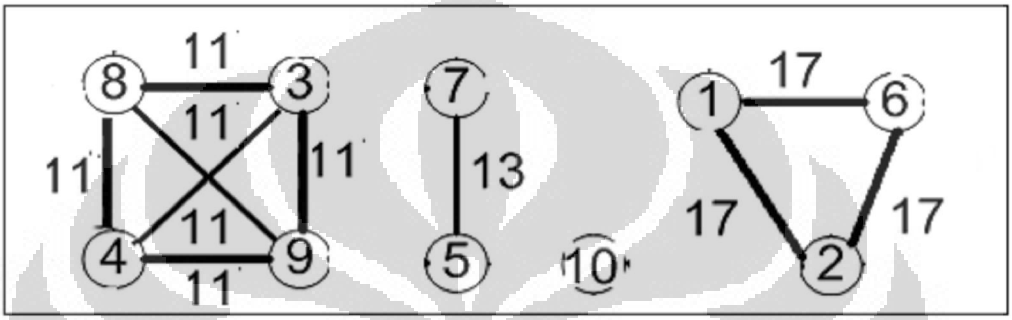
6. Matrik $KR_{|K|x|R|}$, yakni matrik yang mengindikasikan apakah kuliah ke-*i* dapat berlangsung pada ruang ke-*i* dengan melihat kapasitas ruang dan jumlah mahasiswa pengambil kuliah, dimana |K| dan |R| masing-masing adalah jumlah MK dan jumlah ruangan.

   $KR(i, j) = 0,$ jika kuliah ke-*i* tidak dapat dilangsungkan di ruang ke-*j*

   $KR(i, j) = 1,$ jika kuliah ke-*i* dapat dilangsungkan di ruang ke-*j*

7. Daftar kuliah yang dapat dilakukan pada tiap-tiap ruang yang dapat diperoleh dari matrik *KR*.

8. Matrik $KK_{|K|x|K|}$ yakni matrik yang mengindikasikan kuliah-kuliah yang boleh/tidak boleh dilangsungkan pada wakti yang bersamaan. Matrik tersebut dapat diperoleh dari graf kuliah yang diboboti jumlah mahasiswa seperti pada gambar berikut:



Gambar 3: Matrik bentrokan kuliah yang diboboti jumlah mahasiswa

$KK(i, j) = 0,$ jika kuliah ke-*i* dan ke-*j* dapat dilangsungkan pada waktu yang bersamaan.

$KK(i, j) = n,$ jika kuliah ke-*i* dan ke-*j* tidak dapat dilangsungkan pada waktu yang bersamaan, dimana *n* adalah maksimum dari jumlah kelompok mahasiswa pengambil kuliah ke-*i* dan ke-*j*.

Managemen data no 1-5 merupakan data mentah yang diperoleh dari input, interaksi antar komponen tersebut akan menghasilkan data no 6-8 yang akan berperan penting dalam pembuatan jadwal.

3.2.2 Representasi jadwal

Managemen data yang telah disimpan dalam bentuk vektor / matrik digunakan untuk membuat jadwal dimana kumpulan semua jadwal yang

layak berada pada ruang solusi. Representasi jadwal ditulis dalam matrik jadwal $J_{|R|x|W}$, dengan |R| dan |W| masing-masing menyatakan jumlah ruang dan slot waktu dalam seminggu. Elemen dari matrik itu adalah label kuliah dan bilangan -1, jika dalam ruang ke-*i* dan slot-waktu ke-*j* terdapat kuliah maka elemen $J(i, j)$ adalah label kuliahnya, sedangkan jika pada ruang ke-*i* dan slot-waktu ke-*j* tidak ada kuliah maka $j(i, j) = -1$. Sebagai gambaran representasi dari jadwal kuliah perhatikan matrik jadwal berikut:

J	Slot waktu						
Ru	8	9	7	-1			
			-1			4	
an				-1			
			-1				
g		10			20		

Gambar 4: Matrik representasi jadwal *J*

Matrik jadwal *J* akan ditransformasi menjadi kromosom dengan setiap gen membawa informasi mengenai ruang dan slot-waktu masing-masing. Dalam hal ini satu jadwal diwakili oleh satu kromosom, dan tiap kuliah diwakili oleh satu gen, akibatnya dalam kromosom yang terbentuk berukuran sejumlah kuliah yang ada. Kromosom ini merupakan kandidat solusi jadwal kuliah serta memegang peranan penting dalam algoritma memetika.

MK1	MK2	…	…	MKi	…	…	…	MKn
				Ruang				

Gambar 5: Kromosom jadwal sebagai kandidat solusi

3.2.3 Pembuatan populasi awal

Sejumlah n Kromosom sebagai kandidat jadwal awal dibuat, dimana sejumlah n/2 dibuat secara random dan lainnya dibuat secara nonrandom. Pembuatan random cukup dengan menempatkan semua kuliah dalam matrik jadwal J dan elemen matrik jadwal *J* yang masih kosong dinisialisasi dengan -1, sedangkan untuk nonrandom dilakukan dengan menempatkan kuliah demi kuliah ke matrik jadwal *J*. Tiap langkah penjadwalan secara nonrandom, pilih kuliah yang lebih dulu dijadwalkan, kemudian pilih ruang dan slot-waktu untuk menempatkan kuliah tersebut hingga semua kuliah telah terjadwal. Berikut adalah gambaran proses pembuatan jadwal awal secara non random :

1. Definisikan himpunan $\hat{K}$ sebagai himpunan kuliah yang belum dijadwalkan.
2. Pilih kuliah $k \in \hat{K}$ yang mempunyai bentrokan tersesar.
3. Definisikan $\hat{X}$ sebagai himpunan semua posisi yang mungkin untuk menempatkan kuliah *k* dalam matrik jadwal J yang berkontribusi untuk meminimumkan penalti kendala *hard*.

4. Definisikan $\overline{X} \in \hat{X}$, dengan $\overline{X}$ merupakan himpunan semua posisi dalam $\hat{X}$ yang berkontribusi meminimumkan penalti kendala soft.
5. Pilih posisi jadwal *k* dalam $\overline{X}$, keluarkan MK *k* dari $\hat{K}$, lakukan perulangan dari langkah 2 sampai $\hat{K} = \varnothing$.

Pemilihan kuliah dalam langkah 2 juga dapat dilakukan dengan pilih kuliah *k* dengan bentrokan terbesar yang diboboti jumlah mahasiswa, sedangkan untuk langkah 3 dan 4 merupakan cara untuk menentukan posisi untuk kuliah *k*. Pemilihan posisi itu dapat dilakukan dengan pilih slot waktu yang sesuai dengan kehadiran dosen pengajar kuliah *k*, kemudian pilih salah satu ruang kosong yang dapat digunakan untuk melangsungkan kuliah *k.* Oleh karena itu, jika perlu dapat diberikan slot-waktu tambahan sehingga setiap kuliah terjadwal dalam ruang yang sesuai. Setelah jadwal awal terbentuk, akan dilakukan pembentukan populasi kromosom yang memenuhi seluruh kendala *hard* dan dilakukan dengan pencarian lokal.

3.2.4 Solusi kendala hard

Sebuah jadwal yang layak harus memenuhi seluruh kendala *hard* yang berarti penalti kendala *hard* harus bernilai 0. Untuk membangun solusi jadwal yang memenuhi kendala *hard* dilakukan dengan pencarian solusi lokal yakni menukar jadwal 2 kuliah, memindahkan posisi jadwal kuliah ke posisi lain

tanpa mengurangi kelayakan tiap kuliah, atau permutasi 3 jadwal kuliah. Pencarian solusi lokal dilakukan secara berulang sampai setiap kuliah tidak melanggar kendala *hard*.

Setelah seluruh jadwal memenuhi seluruh kendala *hard*, seluruh jadwal ditranformasi ke dalam bentuk kromosom yang terkumpul dalam populasi. Seluruh kromosom diurutkan berdasarkan kualitas yang dapat dilihat dari penalti kendala *soft* tiap-tiap kromosom. Kromosom ini akan mendekati solusi jadwal dengan optimal melalui optimasi kendala *soft*.

3.2.5 Optimasi kendala soft

Optimasi kendala *soft* merupakan bagian terpenting dalam penjadwalan kuliah, sedangkan dalam algoritma memetika proses ini disebut sebagai evolusi / pergantian populasi. Evolusi merupakan pembentukan populasi baru yang beranggotakan kromosom-kromosom yang lebih berkualitas sehingga kandidat solusi akan semakin medekati optimal. Proses evolusi dilakukan dalam ruang solusi jadwal layak (*steady-states process*), yakni hanya kromosom yang tidak melanggar kendala *hard* yang dapat berevolusi serta hanya satu anak yang dapat dihasilkan dalam setiap generasi.

Berikut adalah proses evolusi kromosom dalam mendekati kandidat solusi yang diharapkan.

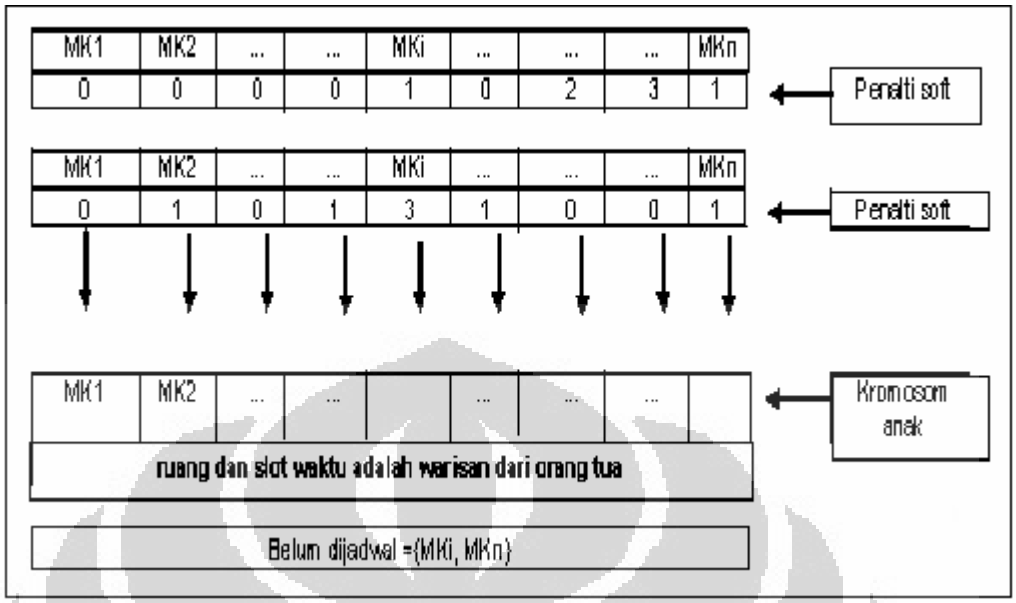
1. Seleksi orang tua

   Proses seleksi dilakukan dengan turnamen ukuran 2, yakni dua

kromosom dipilih secara random dari populasi dan dibandingkan, kromosom paling berkualitas menjadi orang tua ke-1. Pemilihan kromosom sebagai orang tua ke-2 dilakukan dengan cara yang sama.

2. Pindah silang

   Kedua kromosom orang tua terpilih dikombinasikan dengan pindah silang seragam (*uniform crossover*). Tiap-tiap gen yang sama dibandingkan, jika terdapat gen yang tidak terkena penalti kendala *soft* maka gen tersebut akan menjadi anak tanpa melanggar kendala *hard*. Jika gen dari kedua orangtua melanggar kendala *soft* maka gen tersebut dimasukkan ke dalam himpunan Belum dijadwal yang berisi kuliah yang berada di luar jadwal *J*, dimana seluruh anggota dalam himpunan itu akan dijadwalkan kembali dengan bantuan teknik pencarian solusi lokal. Berikut adalah gambar dari pindah silang :



Gambar 6: Pindah silang dengan *uniform crossover*

3. Operator mutasi

   Sebelum penjadwalan kembali himpunan Belum dijadwal, jadwal sementara dilakukan mutasi terhadap perbedaan orang tua. Mutasi dilakukan secara adaptif dengan probabilitas mutasi 0.8 yang dilakukan menukar 2 jadwal kuliah, mengalokasikan kuliah ke dalam ruang dan slot waktu lain, atau permutasi 3 kuliah. Mutasi ini tidak melibatkan kuliah-kuliah himpunan Belum dijadwal.

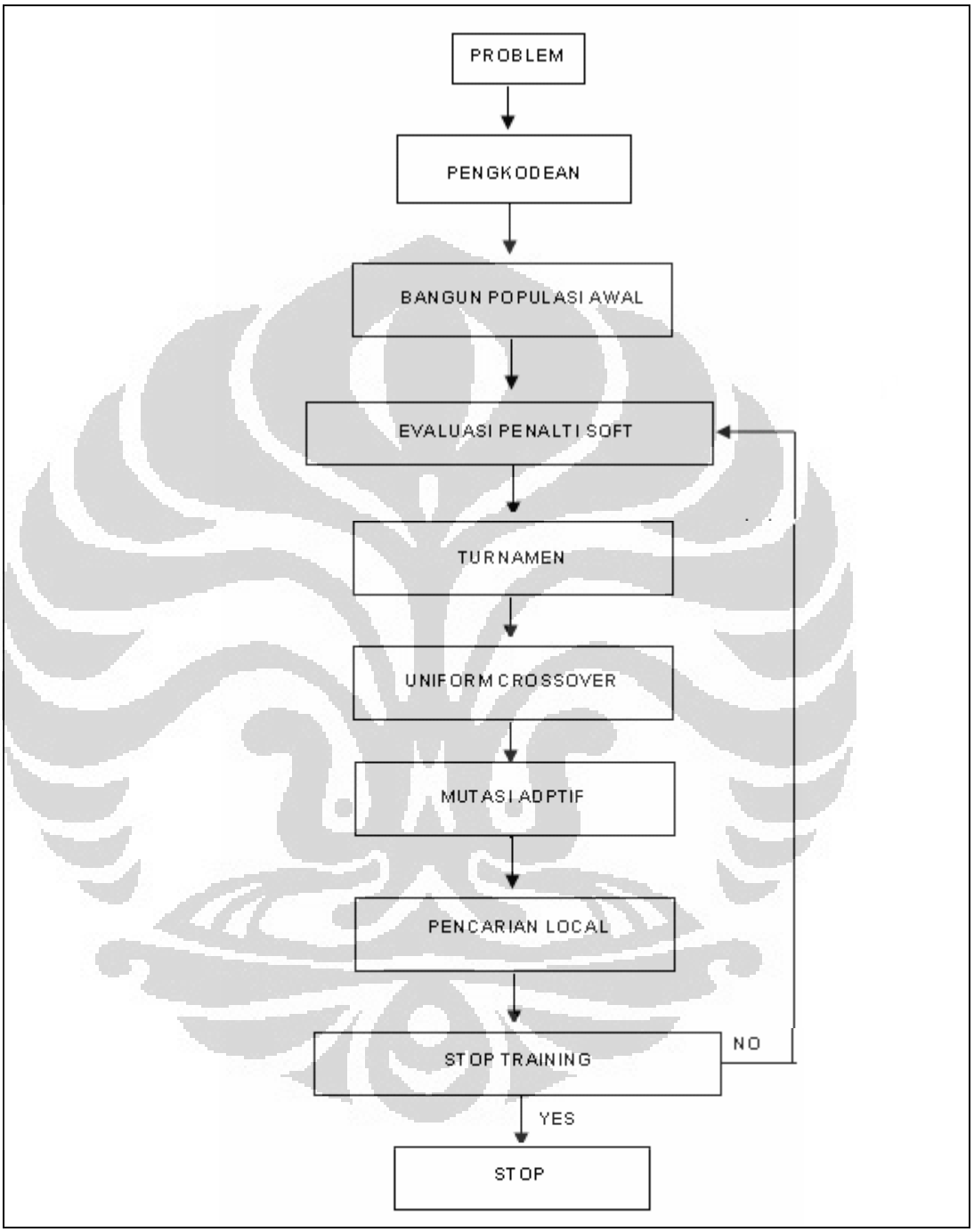
4. Pencarian lokal evolusi

   Pencarian lokal ini dilakukan seperti pencarian lokal pada inisialisasi jadwal. Pencarian lokal evolusi bertujuan menjadwalkan kembali himpunan Belum dijadwal hingga diperoleh jadwal yang memenuhi seluruh kendala *hard*.

5. Pergantian populasi

   Jika semua kuliah sudah terjadwalkan kembali maka jadwal ini akan ditranformasi menjadi kromosom anak yang akan dibandingkan dalam pergantian populasi. Penalti kendala *soft* kromosom anak dihitung kemudian dibandingkan dengan penalti *soft* pada populasi awal, jika maksimum penalti *soft* kromosom dalam populasi melebihi penalti *soft* kromosom anak, maka gantilah kromosom terburuk dengan kromosom anak untuk mendapatkan populasi baru. Proses ini diulang hingga diperoleh kromosom dengan kualitas yang paling baik (koromsom yang optimal).

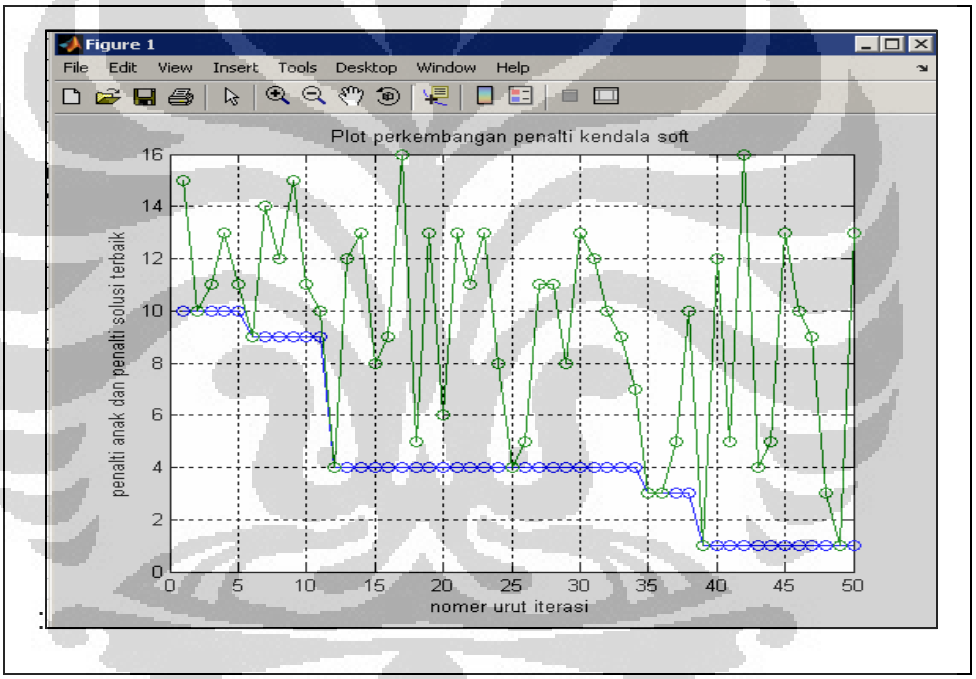
Kromosom dengan kualitas optimal ditransformasi menjadi jadwal yang dapat dibaca oleh pengguna jadwal. Berikut merupakan flowchart penyelesaian penjadwalan kuliah dengan menggunakan algoritma memetika:



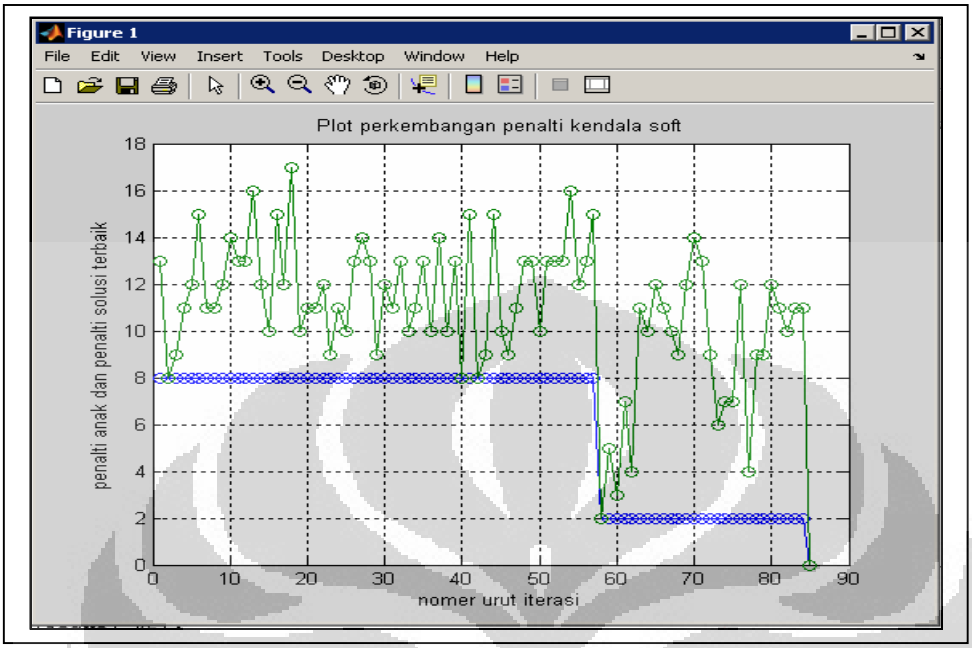
Gambar 7: Flowchart penyelesaian penjadwalan kuliah

3.3 HASIL PERCOBAAN

Algoritma memetika ini diujicobakan untuk masalah penjadwalan kuliah di departemen Matematika UI pada semester genap tahun 2008. Hasilnya adalah seluruh kendala *hard* dapat terpenuhi, begitupun penalti kendala *soft* yang mendekati optimal. Hasil dibawah ini merupakan jadwal dengan menggunakan 8 ruangan, proses evolusi sampai 50 dan 100 generasi yang disertai dengan grafik perkembangan nilai penalti *soft*:



Gambar 8: Grafik penalti *soft* untuk 50 iterasi



Gambar 9: Grafik penalti *soft* untuk 100 iterasi

Diujikan data yang sama dengan menggunakan 5, 6 dan 7 ruang kuliah, hasilnya adalah jadwal yang layak jika digunakan 6 dan 7 ruang, tetapi tidak ditemukan jadwal layak dengan menggunakan 5 ruangan.

Tabel 1: Hasil percobaan untuk 6, 7 dan 8 ruang

Ruang dan iterasi	Waktu total (detik)	Penalti *soft*	Penalti *hard*
6 dan 50	54.375	1	0
6 dan 100	66.6719	2	0
7 dan 50	42.5469	2	0
7 dan 100	51.6719	0	0

8 dan 50	51.1875	1	0
8 dan 100	60.875	0	0
Rata2	54.5547		

Ket: Untuk jumlah ruang 5 tidak ditemukan jadwal yang layak

BERIKUT ADALAH HASIL DARI PENJADWALAN 50 ITERASI

	RUANG	D 108	D 109	D 207	D 302
senin	7.30-9.10	Aljabar Linier Elementer		Topik Khusus II (Digital Image Processing) (K)	
	9.15-10.55	Aljabar I			Topik Khusus I (Analisa Data) (S)
	10.55-12.35			Pemrograman Linier (OR)	PDP & Syarat Batas (M)
	13.15-15.05		Statistika Pengendalian Mutu (S)	Pemrograman Dinamik (OR)	
	15.15-16.55				
selasa	7.30-9.10				
	9.15-10.55	Model Linier II (S)			
	10.55-12.35	Distribusi Loss (A)			
	13.15-15.05				Pemrograman Dinamik (OR)
	15.15-16.55				
rabu	7.30-9.10	Analisis II			
	9.15-10.55	Runtun Waktu (S/A)			
	10.55-12.35				Topik Khusus II (Graph Labeling) (M/K)
	13.15-15.05		Matematika Aktuaria I (A)	Metode Statistika Peubah Ganda (S)	
	15.15-16.55				
kamis	7.30-9.10				
	9.15-10.55	Geometri			
	10.55-12.35	Matematika Diskrit I	Optimisasi pada Jaringan (OR)		
	13.15-15.05				Teori Komputasi (K)
	15.15-16.55				
jumat	7.30-9.10	Statistika Elementer	Pemrograman Linier (OR)		Matematika Numerik I (K)
	9.15-10.55	Aljabar I		Metode Statistika Peubah Ganda (S)	Teori Komputasi (K)
	10.55-12.35				
	13.15-15.05				

	Ruang	D 307	D 401	D 403	D 404
senin	7.30-9.10				
	9.15-10.55	Matematika Aktuaria I (A)			
	10.55-12.35	Komputasi Saintifik (K)			
	13.15-15.05	Model Linier II (S)	Metode Numerik	Geometri	
	15.15-16.55				
selasa	7.30-9.10				
	9.15-10.55				
	10.55-12.35	Runtun Waktu (S/A)			
	13.15-15.05		Matematika Dasar II	Analisis I	Topik Khusus II (Graph Labeling) (M/K)
	15.15-16.55		Analisis II	Teori Ukur dan Integrasi (M/S)	
rabu	7.30-9.10				
	9.15-10.55		Komputasi Saintifik (K)	Matematika Dasar II	Teori Ekonomi Keuangan (A)
	10.55-12.35	Struktur Data dan Algoritma (K)	Matematika Dasar IV	Statistika Elementer	
	13.15-15.05	Teori Ukur dan Integrasi (M/S)			
	15.15-16.55		Metode Numerik	Statistika Matematika II	
kamis	7.30-9.10				
	9.15-10.55		Matematika Dasar IV	Aljabar Linier Elementer	Topik Khusus II (Digital Image Processing) (K)
	10.55-12.35	Matematika Numerik I (K)			PDP & Syarat Batas (M)
	13.15-15.05		Struktur Data dan Algoritma (K)	Analisis I	Topik Khusus I (Analisa Data) (S)
	15.15-16.55		**Statistika Matematika II**		
jumat	7.30-9.10			Statistika Pengendalian Mutu (S)	
	9.15-10.55		Optimisasi pada Jaringan (OR)		Distribusi Loss (A)
	10.55-12.35				
	13.15-15.05				
	TAMPAK BAHWA MK STATISTIKA MATEMATIKA II MASIH DIJADWALKAN PADA SORE HARI				

BERIKUT ADALAH HASIL DARI PENJADWALAN 100 ITERASI

ruang		D 108	D 109	D 207	D 302
senin	7.30-9.10				Topik Khusus II (Digital Image Processing) (K)
	9.15-10.55		Statistika Pengendalian Mutu (S)		
	10.55-12.35	Matematika Dasar IV			
	13.15-15.05		Komputasi Saintifik (K)		
	15.15-16.55				
selasa	7.30-9.10	Metode Numerik	Runtun Waktu (S/A)	Optimisasi pada Jaringan (OR)	
	9.15-10.55				Pemrograman Dinamik (OR)
	10.55-12.35				Pemrograman Linier (OR)
	13.15-15.05				Topik Khusus II (Graph Labeling) (M/K)
	15.15-16.55				
rabu	7.30-9.10	Runtun Waktu (S/A)			
	9.15-10.55		Komputasi Saintifik (K)		
	10.55-12.35				Topik Khusus II (Graph Labeling) (M/K)
	13.15-15.05	Analisis II		Teori Ekonomi Keuangan (A)	Topik Khusus II (Digital Image Processing) (K)
	15.15-16.55				
kamis	7.30-9.10	Distribusi Loss (A)		Matematika Numerik I (K)	
	9.15-10.55			Topik Khusus I (Analisa Data) (S)	
	10.55-12.35		Matematika Aktuaria I (A)		
	13.15-15.05		Pemrograman Linier (OR)		
	15.15-16.55				
jumat	7.30-9.10	Aljabar Linier Elementer	Struktur Data dan Algoritma (K)		
	9.15-10.55			PDP & Syarat Batas (M)	
	10.55-12.35	Metode Numerik		Teori Komputasi (K)	
	13.15-15.05		Optimisasi pada Jaringan (OR)	Matematika Aktuaria I (A)	Matematika Numerik I (K)

ruang		D 307	D 401	D 403	D 404
senin	7.30-9.10	Teori Ukur dan Integrasi (M/S)			
	9.15-10.55		Statistika Matematika II	PDP & Syarat Batas (M)	

	10.55-12.35	Distribusi Loss (A)			Model Linier II (S)
	13.15-15.05				
	15.15-16.55				
selasa	7.30-9.10			Geometri	
	9.15-10.55	Struktur Data dan Algoritma (K)	Model Linier II (S)	Matematika Dasar IV	
	10.55-12.35		Statistika Pengendalian Mutu (S)	Analisis II	Teori Komputasi (K)
	13.15-15.05		Analisis I	Matematika Dasar II	
	15.15-16.55				
rabu	7.30-9.10	Statistika Matematika II			
	9.15-10.55	Statistika Elementer	Pemrograman Dinamik (OR)	Aljabar I	
	10.55-12.35			Statistika Elementer	Topik Khusus I (Analisa Data) (S)
	13.15-15.05				Metode Statistika Peubah Ganda (S)
	15.15-16.55				
kamis	7.30-9.10	Teori Ukur dan Integrasi (M/S)	Geometri		
	9.15-10.55		Aljabar Linier Elementer		
	10.55-12.35				
	13.15-15.05		Metode Penelitian	Matematika Dasar II	
	15.15-16.55				
jumat	7.30-9.10		Analisis I		
	9.15-10.55		Aljabar I	Matematika Diskrit I	Metode Statistika Peubah Ganda (S)
	10.55-12.35				
	13.15-15.05				
	15.15-16.55				

TAMPAK BAHWA JADWAL MENCAPAI OPTIMAL

BAB IV

ANALISIS HASIL

Dengan menggunakan 45 slot-waktu dan 8 ruang kuliah, diperoleh jadwal yang sesuai dengan batasan dan tujuan yang diharapkan. Beberapa kali running menghasilkan jadwal yang sama sekali tidak melanggar kendala *hard*, walaupun kadang jadwal yang yang dihasilkan masih mengandung sedikit pelanggaran kendala *soft* pada iterasi kurang dari 50. Namun, sebelum iterasi ke 100 jadwal telah memenuhi seluruh kendala *soft*.

Analisis grafik penalti kendala *soft* tiap iterasi dalam evolusi sangatlah penting dalam melihat kinerja program. Terlihat dalam grafik penalti *soft* tiap generasi cenderung tidak naik, artinya kualitas jadwal cenderung meningkat mendekati jadwal dengan kualitas optimal.

Dari gambar 8 tampak bahwa sampai iterasi ke-50 nilai penalti adalah 1, artinya terdapat 1 kendala *soft* dari suatu kuliah yang tidak terpenuhi. Hal ini dapat dilihat dari jadwal 50 iterasi yakni kuliah ”Statistika Matematika 2” dijadwalkan pada sore hari. Namun, terlihat dari gambar 9 pada iterasi ke-83 nilai penalti *soft* adalah 0, artinya seluruh kendala soft terpenuhi.

Titik-titik diatas garis penalti *soft* merupakan nilai dari penalti soft kromosom anak dalam setiap generasi. Penalti *soft* kromosom anak belum tentu menjadi yang terbaik dalam populasi, tetapi kromosom anak

mendukung untuk menghasilkan populasi yang lebih berkualitas untuk generasi selanjutnya.

Kelebihan memetika dalam penjadwalan kuliah ini adalah memungkinkan untuk mencapai solusi optimal global. Artinya, jadwal yang terbentuk dapat mencapai sedekat mungkin dengan solusi yang diharapkan. Namun, metode ini juga memiliki kelemahan yaitu membutuhkan waktu komputasi lebih lama dibandingkan dengan algoritma genetika.

BAB V

KESIMPULAN

Dengan algoritma memetika diperoleh hasil jadwal yang sedekat mungkin dengan solusi jadwal optimal. Dari uji coba 50 iterasi seluruh kendala *hard* terpenuhi, dan hanya satu kuliah yang tidak memenuhi kendala *soft*. Namun, dengan 100 iterasi diperoleh jadwal kuliah yang memenuhi seluruh kendala *hard* dan *soft* dengan running yang kurang dari 2 menit. Dengan demikian algoritma memetika dapat digunakan untuk menyelesaiakan penjadwalan kuliah dengan efektif.

Penjadwalan kuliah dengan algoritma memetika dapat menghasilkan jadwal sesuai dengan data dan kendala serta prioritas kendala. Jadwal kuliah yang dihasilkan dapat memenuhi semua kendala *hard* dapat dikatakan jadwal yang layak. Sedangkan kuliah yang tidak dapat memenuhi semua kendala soft, melainkan hanya sebagian kendala saja yang terpenuhi, maka kuliah ini akan ditempatkan pada slot yang mempunyai nilai optimal atau dengan nilai penalti *soft* terendah.

DAFTAR PUSTAKA

Abdullah, Salwani., Burke, Edmund K., and McKullum, Barry.2007. *"A Hybrid Evolutionary Approach to the University Course Timetabling Problem".*

Alkan, Alpay.2003.*"Memetic Algorithm for Timetabling".* Departement of Computer Enginering Yeditepe University

Burke, E.K and Silva, JD Landa. 2004.*"The design of memetics Algorithm for scheduling and Timetabling problems".* The University of Nottingham, UK

Elfitriadi, Epa.2001.*"Application Simulated anneling of school timetabling".* Departement Mathematics University of Indonesia

Herlina. 2004."Penyelesaian Masalah Penjadwalan Kuliah Menggunakan Pewarnaan Simpel Graf dengan Metode Tabu Search". Bandung Institute of Technology

Lourenco, H. R., O. Martin, and T. Stutzle.2002.*"Iterated Local Search, Handbook of Metaheurisitcs,*321-353,*"*.KluwerAcademic Publishers, Norwell, MA, USA

Rossidoria, Olivia, dkk. 2004.*"An Effective Hybrid Algorithm for University Course Timetabling".* School of Computing Napier University

Rossidoria, Olivia. 2004*."A Memetic Algorithm for University Course Timetabling".* School of Computing Napier University

IMPLEMENTASI PROGRAM DENGAN SOFTWARE MATLAB 7.1

```
clc;clear;
channel = ddeinit('excel','KULIAH_MAHASISWA.xls');
KS = ddereq(channel,'r3c5:r63c11')
jumlah_mhs_k = JUMLAH_MHS_K(KS);
transpose(jumlah_mhs_k);
KR = KULIAH_RUANG(jumlah_mhs_k);
kuliahdiR= KULIAH_DI_R(KR);%banyaknya kuliah yang di ruang R
KK = MATRIK_ADJENSI(KS);
korelasi = KORELASI(KK);
slot=input('masukkan slot :');
%inisialisasi solusi
clc;
for n=1:1:5
   [jadwal,fungsi_eval_1,kontrol]=PENCARIAN_LOKAL(KR, KK,slot);
   if n==1
      A=jadwal
      nilai=FITNESS(jadwal,KS);
   end
   if n==2
      B=jadwal
      nilai=FITNESS(jadwal,KS);
   end
   if n==3
      C=jadwal
      nilai=FITNESS(jadwal,KS);
```

```
    end
    if n==4
        D=jadwal
        nilai=FITNESS(jadwal,KS);
    end
    if n==5
        E=jadwal
        nilai=FITNESS(jadwal,KS);
    end
%menghitung fungsi evaluasi1 untuk hard konstrian solver
    penalti(:,n)=[fungsi_eval_1,nilai];
    kontrol;
end
for n=1:1:5
    [jadwal,fungsi_eval_1,kontrol]=PENCARIAN_LOKAL_RANDOM(KR,KK,slot);
    if n==1
        F=jadwal
        nilai=FITNESS(jadwal,KS);
    end
    if n==2
        G=jadwal
        nilai=FITNESS(jadwal,KS);
    end
    if n==3
        H=jadwal
        nilai=FITNESS(jadwal,KS);
    end
```

```
    if n==4
        I=jadwal
        nilai=FITNESS(jadwal,KS);
    end
    if n==5
        J=jadwal
        nilai=FITNESS(jadwal,KS);
    end
%menghitung fungsi evaluasi1 untuk hard konstrian solver
    penalti(:,5 + n)=[fungsi_eval_1,nilai];
    kontrol;
end

    %berikut adalah fungsi evolusi
    himpunan(1,1)=FITNESS(A,KS);
    himpunan(1,2)=FITNESS(B,KS);
    himpunan(1,3)=FITNESS(C,KS);
    himpunan(1,4)=FITNESS(D,KS);
    himpunan(1,5)=FITNESS(E,KS);
    himpunan(1,6)=FITNESS(F,KS);
    himpunan(1,7)=FITNESS(G,KS);
    himpunan(1,8)=FITNESS(H,KS);
    himpunan(1,9)=FITNESS(I,KS);
    himpunan(1,10)=FITNESS(J,KS);
iterasi=1
while iterasi<=50
    [anak, jadwal,nilai, penalti]=EVOLUSI(A,B,C,D,E,F,G,H,I,J,KS,KR,KK,slot);
```

```
X(iterasi)=iterasi;
Z(iterasi)=nilai;
if (max(himpunan(1,:)))> nilai
   for counter=1:10
      if himpunan(1,counter)==max(himpunan)
         individu=counter;
         break;
      end
   end
   switch individu>0
      case individu==1;
         A=anak;
         himpunan(1,1)=nilai;
      case individu==2;
         B=anak;
         himpunan(1,2)=nilai;
      case individu==3;
         C=anak;
         himpunan(1,3)=nilai;
      case individu==4;
         D=anak;
         himpunan(1,4)=nilai;
      case individu==5;
         E=anak;
         himpunan(1,5)=nilai;
      case individu==6;
         F=anak;
```

```
            himpunan(1,6)=nilai;
        case individu==7;
            G=anak;
            himpunan(1,7)=nilai;
        case individu==8;
            H=anak;
            himpunan(1,8)=nilai;
        case individu==9;
            I=anak;
            himpunan(1,9)=nilai;
        otherwise individu==10;
            J=anak;
            himpunan(1,10)=nilai;
    end
else
    ;
end
%solusi sudah optimal
if nilai==0;
    solusi=anak;
    break;
end
Y(iterasi)=min(himpunan(1,:));
iterasi=iterasi+1;
end
himpunan
clc;
```

```
for count=1:10
   if himpunan(1,count)==min(himpunan)
      individu=count;
      break;
   end
end
switch individu>0 & nilai>0
          case individu==1;
            solusi=A;
          case individu==2;
          solusi=B;
          case individu==3;
          solusi=C;
         case individu==4;
          solusi=D;
         case individu==5;
          solusi=E;
         case individu==6;
          solusi=F;
         case individu==7;
          solusi=G;
         case individu==8;
          solusi=H;
         case individu==9;
          solusi=I;
          otherwise individu==10;
            solusi=J;
```

```
end
jumlah_sks=max(solusi);
solusi=solusi(:,1:slot)
gen_jadwal=transpose(GEN_JADWAL(solusi));
gen_jadwal
[f1,f2,f3]=PENALTI(solusi,KS)
f=min(himpunan)
g=sum(FUNGSI_EVALUASI1(KR,KK, solusi))
iterasi-1
plot(X,Y,'-o',X,Z,'-o'),grid
xlabel('nomer urut iterasi')
ylabel('penalti anak dan penalti solusi terbaik')
title('Plot perkembangan penalti kendala soft')

function [anak_sementara, belum_dijadwal]=crossover(parent1, parent2, KR, KK, KS)
%program ini untuk mencari generasi dari kedua orang tua terpilih
[jumlah_ruang, jumlah_slot]=size(parent1);
anak=-ones(jumlah_ruang, jumlah_slot);
kromosom1=transpose(GEN_JADWAL(parent1));
g_anak1=FUNGSI_EVALUASI1(KR, KK, parent1);
[f1,f2,f3]=PENALTI(parent1,KS);
f_anak1=(f1+f2)+f3;
kromosom2=transpose(GEN_JADWAL(parent2));
g_anak2=FUNGSI_EVALUASI1(KR,KK, parent2);
[f1,f2,f3]=PENALTI(parent2,KS);
f_anak2=(f1+f2)+f3;
[baris_gen, jumlah_gen]=size(kromosom1);
```

```
kosong=0;
belum_dijadwal=-1;
for kontrol=1:jumlah_gen
   switch kontrol==kontrol
      case (g_anak1(1,kontrol)==0)& (f_anak1(1,kontrol)<=f_anak2(1,kontrol))
         slot  = kromosom1(1,kontrol);
         ruang = kromosom1(2,kontrol);
         if anak(ruang, slot)==-1
            anak(ruang, slot)=kontrol;
         else
            kosong=kosong+1;
            belum_dijadwal(1,kosong)=kontrol;
         end
      case (g_anak2(1,kontrol)==0)& (f_anak2(1,kontrol)<=f_anak1(1,kontrol));
         slot  = kromosom2(1,kontrol);
         ruang = kromosom2(2,kontrol);
         if anak(ruang, slot)==-1
            anak(ruang, slot)=kontrol;
         else
            kosong=kosong+1;
            belum_dijadwal(1,kosong)=kontrol;
         end
      case (g_anak2(1,kontrol)>0)&(g_anak2(1,kontrol)>0)
         kosong=kosong+1;
         belum_dijadwal(1,kosong)=kontrol;
      otherwise
         ;
```

```
    end
end
anak_sementara=anak;


function  [anak, jadwal,nilai, penalti]=EVOLUSI(A,B,C,D,E,F,G,H,I,J,KS,KR,KK,slot)
%program ini digunakan untuk meningkatkan kualitas jadwal
%berikut adalah proses seleksi individu untuk menjadi orang tua
seleksi(1,1)=floor(10*rand(1))+1;
seleksi(1,2)=floor(10*rand(1))+1;
seleksi(1,3)=floor(10*rand(1))+1;
seleksi(1,4)=floor(10*rand(1))+1
for kontrol=1:4
    switch kontrol==kontrol;
        case seleksi(1,kontrol)==1
            calon=A;
        case seleksi(1,kontrol)==2
            calon=B;
        case seleksi(1,kontrol)==3
            calon=C;
        case seleksi(1,kontrol)==4
            calon=D;
        case seleksi(1,kontrol)==5
            calon=E;
        case seleksi(1,kontrol)==6
            calon=F;
        case seleksi(1,kontrol)==7
            calon=G;
```

```
        case seleksi(1,kontrol)==8
            calon=H;
        case seleksi(1,kontrol)==9
                calon=I;
        otherwise seleksi(1,kontrol)==10 ;
            calon=J;
    end
    switch kontrol==kontrol
        case kontrol==1
            individu1=calon;
            nilai1=FITNESS(calon,KS);
            f1=sum(FUNGSI_EVALUASI1(KR, KK, calon));
        case kontrol==2
            individu2=calon;
            nilai2=FITNESS(calon,KS);
            f2=sum(FUNGSI_EVALUASI1(KR, KK, calon));
        case kontrol==3;
            individu3=calon;
            nilai3=FITNESS(calon,KS);
            f3=sum(FUNGSI_EVALUASI1(KR, KK, calon));
        otherwise
            individu4=calon;
            nilai4=FITNESS(calon,KS);
            f4=sum(FUNGSI_EVALUASI1(KR, KK, calon));
    end
end
penalti(1,1:4)=[f1,f2,f3,f4];
```

```
penalti(2,1:4)=[nilai1,nilai2,nilai3,nilai4];
%seleksi kedua orangtua berdasarkan fitness
if ((penalti(2,1)<penalti(2,2)) & (penalti(1,1)==0))
   parent1=individu1;
else parent1=individu2;
end
if ((penalti(2,3)<penalti(2,4)) & (penalti(1,3)==0))
   parent2=individu1;
else parent2=individu2;
end
[anak_sementara, belum_dijadwal]=crossover(parent1, parent2, KR, KK, KS)
[anak, fungsi_eval1, kontrol]=PENCARIAN_LOKAL_EVOLUSI(KS, anak_sementara,
belum_dijadwal, KR, KK, slot);
jadwal=belum_dijadwal;
nilai=FITNESS(anak,KS);


function nilai=FITNESS(jadwal,KS)
%fungsi ini menghitung nilai fungsi evaluasi kendala soft tiap2
%individu/jadwal
[jumlah_ruang, jumlah_slot]=size(jadwal);
nilai=0;
sementara=zeros(1,7);
[jumlah_gen, mahasiswa]=size(KS);
f1=zeros(1,jumlah_gen);
f2=zeros(1,jumlah_gen);
f3=zeros(1,jumlah_gen);
for slot=1:1:jumlah_slot
```

```
for ruang=1:1:jumlah_ruang
    pengontrol=zeros(1,7);
    %hitung S3
    kuliah=jadwal(ruang, slot);
    for mahasiswa=1:1:7
        if kuliah > 0 & KS(kuliah,mahasiswa)==1
            pengontrol(1,mahasiswa)=pengontrol(1,mahasiswa)+1;
            if pengontrol(1,mahasiswa)>3
                f3(1,kuliah)=f3(1,kuliah)+1;
            end
        end
    end
    %hitung nilai dari kendala soft ke-1
    if mod(slot,5)==0 & jadwal(ruang, slot)>0
        kuliah=jadwal(ruang, slot);
        f1(1,kuliah)=1;
    end
    %hitung nilai dari S2
    if kuliah<0
        ;
    else kontrol=slot;
        while (mod(kontrol,5)>0 & kontrol<(jumlah_slot+1))
            waktu=kontrol;
            for ruangan=1:1:jumlah_ruang
                if (((jadwal(ruangan, waktu)-kuliah) == (31))|((jadwal(ruangan, waktu)-kuliah) ==
-((31))))
                    f2(1,kuliah)=1;break;
```

```
            end
          end
          kontrol=kontrol+1;
        end
      end
    end
end
nilai=sum(f3)+sum(f1)+sum(f2);


function g = FUNGSI_EVALUASI1(KR, KK, jadwal)
%fungsi ini menghitung nilai fungsi evaluasi tiap kuliah
[jumlah_sks, jumlah_ruang]=size(KR);
gen_jadwal=GEN_JADWAL(jadwal);
slot=max(gen_jadwal(:,1));
jadwal=-ones(jumlah_ruang, slot);
for i=1:1:jumlah_sks
   waktu=gen_jadwal(i,1);
   ruang=gen_jadwal(i,2);
   diisi=i;
   jadwal(ruang, waktu)=diisi;
end
for i=1:1:jumlah_ruang
   for j=1:1:slot
      if jadwal(i,j)>0
         k=jadwal(i,j);
         gk2(1,k)=0;
         if KR(k,i)==0    %buat fungsi g2
```

```
            gk2(1,k)=1;
        end
        %buat fungsi g3
        gk3(1,k)=0;
        for ruang=1:1:jumlah_ruang
            if jadwal(ruang,j)>0
                l=jadwal(ruang,j);
                if KK(k,l)>0
                    gk3(1,k)=gk3(1,k)+1;
                end
            end
        end
        gk(1,k)=gk2(1,k)+gk3(1,k);
    end
  end
end %fungsi evaluasi diminimumkan sampai dengan 0 menggunakan N1 dan N2
g=gk;


function gen_jadwal=GEN_JADWAL(jadwal);
%fungsi ini merubah jadwal menjadi mtrik jumlah_sks kali 2
%kolom pertama menyatakan slot dari masing-kuliah
%kolom kedua menyatakan ruangannya
[jumlah_ruang slot]=size(jadwal);
for i=1:1:jumlah_ruang
  for j=1:1:slot
    if jadwal(i,j)>0
        diisi=jadwal(i,j);
```

```
            gen_jadwal(diisi,1)=j;
            gen_jadwal(diisi,2)=i;
        else
            ;
        end
    end
end


function [jadwal]=INISIAL_HEURISTIK(KR, KK)
%fungsi untuk membuat jadwal pertama kali
[jumlah_sks, jumlah_ruang]=size(KR);
kuliahdiR=KULIAH_DI_R(KR);
korelasi = KORELASI(KK);
HE=(1:1:jumlah_sks);
jadwal=-ones(jumlah_ruang, jumlah_sks);
waktu_mak=0;
while max(HE)>0
    for i=1:1:jumlah_sks
        %pilih kuliah dengan jumlah bentrokan maksimum
        if korelasi(i)==max(korelasi)
            korelasi(i)=-1;
            for ruang=1:1:jumlah_ruang
                if kuliahdiR(ruang)==min(kuliahdiR)
                    kuliahdiR(ruang)=kuliahdiR(ruang)+1;
                    diisi=ruang;
                    for waktu=1:1:jumlah_sks
                        if jadwal(diisi,waktu)<1 & KR(i,diisi)>0
```

```
                jadwal(diisi,waktu)=i;
                HE(i)=-1;
                if waktu_mak > waktu;
                    ;
                else
                    waktu_mak=waktu;
                end
                break;
              end
            end
          end
          if HE(i)==-1
            break;
          end
        end
      end
    end
end
jadwal=jadwal(:,1:waktu_mak);

function [jadwal]=INISIAL_RANDOM(KR, KK, slot)
%program ini membuat jadwal awal secara random
[jumlah_sks, jumlah_ruang]=size(KR);
HE=(1:1:jumlah_sks);
jadwal=-ones(jumlah_ruang, slot);
for i=1:1:jumlah_sks
  while HE(i)>0
```

```
        baris=floor(jumlah_ruang*rand(1))+1;
        kolom=floor(slot*rand(1))+1;
        if jadwal(baris, kolom)==-1
            jadwal(baris, kolom)=i;
            HE(i)=-1;
        end
    end
end


function [jadwal,g,kontrol]=INISIALISASI_JADWAL(KRKK)
%berikut akan dibuat matrik isialisasi jadwal J ukuran ruangan kali 45
[jumlah_sks, jumlah_ruang_plusK]=size(KRKK);
jumlah_ruang=jumlah_ruang_plusK-jumlah_sks;
KR=KRKK(:,1:jumlah_ruang);
KK=KRKK(:,jumlah_ruang+1:jumlah_ruang_plusK);
jadwal_awal=JADWAL_AWAL(KRKK);
[slot,jadwal]=PENYESUAIAAN_SLOT(jadwal_awal);
gen_jadwal=GEN_JADWAL(jadwal);
KRKK_gen_jadwal=[KRKK gen_jadwal];
g = FUNGSI_EVALUASI1(KRKK_gen_jadwal);
kontrol=0;

while kontrol<100000 & sum(g)>0 %pilih yang maksimum
    kuliah=sum(floor(2*rand(1,jumlah_sks-1)))+1;
    for k=1:1:jumlah_ruang
        g_akhir=zeros(1,jumlah_sks);
        for l=1:1:slot
```

```
        if jadwal(k,l)==kuliah
            baris=floor(jumlah_ruang*rand(1))+1;
      kolom=floor(slot*rand(1))+1;
            jadwal(k,l)=jadwal(baris,kolom);
            jadwal(baris,kolom)=kuliah;
            %berikut adalah fungsi evaluasi
            gen_jadwal=GEN_JADWAL(jadwal);
            KRKK_gen_jadwal=[KRKK gen_jadwal];
            g_akhir = FUNGSI_EVALUASI1(KRKK_gen_jadwal);
            %akhir dari fungsi evaluasi jadwal
            if sum(g_akhir)<= sum(g)
               g=g_akhir;
               break;
            else
               jadwal(baris,kolom)=jadwal(k,l);
               jadwal(k,l)=kuliah;break;
            end
         end
      end
      if sum(g)==sum(g_akhir)
         break;
      end
   end
    kontrol=kontrol+1;
end
fungsi_eval_1=sum(g);
kontrol
```

```
function [jadwal]=JADWAL_AWAL(KRKK)
%fungsi untuk membuat jadwal tertama kali
[jumlah_sks, jumlah_ruang_plusK]=size(KRKK);
jumlah_ruang=jumlah_ruang_plusK-jumlah_sks;
KR=KRKK(:,1:jumlah_ruang);
KK=KRKK(:,jumlah_ruang+1:jumlah_ruang_plusK);
kuliahdiR=KULIAH_DI_R(KR);
korelasi = KORELASI(KK);
HE=(1:1:jumlah_sks);
jadwal=-ones(jumlah_ruang,jumlah_sks);
waktu_mak=0;
while max(HE)>0
   for i=1:1:jumlah_sks
      if korelasi(i)==max(korelasi)
         korelasi(i)=-1;
        for waktu=1:1:jumlah_sks
           for ruang=1:1:jumlah_ruang
              if kuliahdiR(ruang)==min(kuliahdiR)
                 kuliahdiR(ruang)=kuliahdiR(ruang)+1;
                 diisi=ruang;
                 if jadwal(diisi,waktu)<1 & KR(i,diisi)==1
                    jadwal(diisi,waktu)=i;
                    HE(i)=-1;
                    if waktu_mak > waktu;
                       ;
                    else
```

```
                    waktu_mak=waktu;
                end
                break;
            end
        end
    end
    if HE(i)==-1
    break;
    end
  end
 end
end
jadwal=jadwal(:,1:waktu_mak);

function [jumlah_mhs_k] = JUMLAH_MHS_K(KS)
%input fungsi berupa matrik KS
%output fungsi berupa jumlah mahasiswa yang mengambil tiap-tiap kuliah
[jumlah_sks,jumlah_mhs] = size(KS);
for i=1:1:jumlah_sks
   jumlah_mhs_k(i)=0;
   for j=1:1:jumlah_mhs
      jumlah_mhs_k(i)=jumlah_mhs_k(i)+KS(i,j);
   end
end

function [korelasi] = KORELASI(KK)
```

```
%mendefinisikan matrik korelasi tiap kuliah
[jumlah_sks, jumlah]=size(KK);
korelasi=zeros(1,jumlah_sks);
for i=1:1:jumlah_sks
   for j=1:1:jumlah
      korelasi(1,i)= korelasi(1,i)+ KK(i,j);
   end
end

function kuliahdiR= KULIAH_DI_R(KR);
%banyaknya kuliah yang di ruang R
[jumlah_sks, jumlah_ruang]=size(KR);
for j=1:1:jumlah_ruang
   kuliahdiR(j)=0;
   for i=1:1:jumlah_sks
      if KR(i,j)==1
         kuliahdiR(j)=kuliahdiR(j)+1;
      end
   end
end

function [KR] = KULIAH_RUANG(jumlah_mhs_k )
%output fungsi berupa matrik kuliah cross ruang
%input jumlah ruang untuk kuliah
channel = ddeinit('excel','DATA_KAPASITAS_RUANG_KULIAH.xls');
kapasitas = ddereq(channel,'r5c3:r12c3')
fitur_ruangan =ones(61,8);
```

```
[jumlah_sks,jumlah_ruang] = size(fitur_ruangan);
for i=1:1:jumlah_sks
   for j=1:1:jumlah_ruang
      if fitur_ruangan(i,j)==0
         KR(i,j)=0;
      else if kapasitas(j)>=jumlah_mhs_k(i)
            KR(i,j)=1;
          else
            KR(i,j)=0;
          end
      end
   end
end

function [KK]= MATRIK_ADJENSI(KS)
%mendefinisikan fungsi matrik adjensi antar kuliah
[jumlah_sks,jumlah_mhs] = size(KS)
KK=zeros(jumlah_sks,jumlah_sks);
for i=1:1:jumlah_sks-1
   sementara=i;
   for k=(sementara+1):1:jumlah_sks
      for j=1:1:jumlah_mhs
         if (KS(i,j)>0 & KS(k,j)>0)
            KK(i,k)= KK(i,k)+KS(i,j);
            KK(k,i)=KK(i,k);
         end
      end
```

```
    end
end


function [f1,f2,f3]=PENALTI(jadwal,KS)
%fungsi ini menghitung nilai fungsi evaluasi kendala soft tiap2
%individu/jadwal
[jumlah_ruang, jumlah_slot]=size(jadwal);
nilai=0;
sementara=zeros(1,7);
[jumlah_gen, mahasiswa]=size(KS);
f1=zeros(1,jumlah_gen);
f2=zeros(1,jumlah_gen);
f3=zeros(1,jumlah_gen);
for slot=1:1:jumlah_slot
    for ruang=1:1:jumlah_ruang
        pengontrol=zeros(1,7);
        %hitung S3
        kuliah=jadwal(ruang, slot);
        for mahasiswa=1:1:7
            if kuliah > 0 & KS(kuliah,mahasiswa)==1
                pengontrol(1,mahasiswa)=pengontrol(1,mahasiswa)+1;
                if pengontrol(1,mahasiswa)>3
                    f3(1,kuliah)=f3(1,kuliah)+1;
                end
            end
        end
        %hitung nilai dari kendala soft ke-1
```

```
        if mod(slot,5)==0 & jadwal(ruang, slot)>0
            kuliah=jadwal(ruang, slot);
            f1(1,kuliah)=1;
        end
        %hitung nilai dari S2
        if kuliah<0
            ;
        else kontrol=slot;
            while (mod(kontrol,5)>0 & kontrol<(jumlah_slot+1))
                waktu=kontrol;
                for ruangan=1:1:jumlah_ruang
                    if (((jadwal(ruangan, waktu)-kuliah) == 31)|((jadwal(ruangan, waktu)-kuliah) == -
31))
                        f2(1,kuliah)=1;break;
                    end
                end
                kontrol=kontrol+1;
            end
        end
    end
end

function [jadwal,fungsi_eval_1,kontrol]=PENCARIAN_LOKAL(KR, KK, slot)
%berikut akan dibuat matrik inisialisasi jadwal J ukuran ruangan kali 45
[jumlah_sks, jumlah_ruang]=size(KR);
jadwal_awal=INISIAL_HEURISTIK(KR, KK);
[slot,jadwal]=PENYESUAIAAN_SLOT(jadwal_awal, slot);
```

```
gen_jadwal=GEN_JADWAL(jadwal);
KRKK_gen_jadwal=[KR KK gen_jadwal];
g = FUNGSI_EVALUASI1(KR, KK, jadwal);
kontrol=0;
while kontrol<50000 & sum(g)>0 %pilih yang maksimum
   kuliah=sum(floor(2*rand(1,jumlah_sks-1)))+1;
   for k=1:1:jumlah_ruang
      g_akhir=zeros(1,jumlah_sks);
      for l=1:1:slot
         if jadwal(k,l)==kuliah
            baris=floor(jumlah_ruang*rand(1))+1;
            kolom=floor(slot*rand(1))+1;
            jadwal(k,l)=jadwal(baris,kolom);
            jadwal(baris,kolom)=kuliah;
            %berikut adalah fungsi evaluasi
            gen_jadwal=GEN_JADWAL(jadwal);
            KRKK_gen_jadwal=[KR KK gen_jadwal];
            g_akhir = FUNGSI_EVALUASI1(KR, KK, jadwal);
            g_akhir=sum(g_akhir);
            %akhir dari fungsi evaluasi jadwal
            if sum(g_akhir)<= sum(g)
               g=g_akhir;
               break;
            else
               jadwal(baris,kolom)=jadwal(k,l);
               jadwal(k,l)=kuliah;break;
            end
```

```
        end
      end
      if sum(g)==sum(g_akhir)
        break;
      end
    end
    kontrol=kontrol+1;
end
kontrol
fungsi_eval_1=sum(g);

function [anak, fungsi_eval_1,
kontrol]=PENCARIAN_LOKAL_EVOLUSI(KS,anak_sementara, belum_dijadwal,KR,KK,slot)
%berikut akan dibuat matrik isialisasi jadwal J ukuran ruangan kali 45
[jumlah_sks, jumlah_ruang]=size(KR);
[satu, banyak]=size(belum_dijadwal);
anak_sementara(:,(slot+1):(slot+banyak+1))=-1;
while(max(belum_dijadwal))>0
  for kontrol=1:banyak
    if belum_dijadwal(1,kontrol)==-1
      break;
    else
      for waktu=slot:-1:1
        for ruang=1:jumlah_ruang
          kuliah=belum_dijadwal(1,kontrol)
          if ((anak_sementara(ruang, waktu)<0 )& (kuliah >0)& mod(waktu,5)>0)
            if (KR(kuliah, ruang)==1)
```

```
                    anak_sementara(ruang, waktu)=kuliah;
                    belum_dijadwal(1,kontrol)=-1;break;
                end
            end
        end
        %untuk keluar dari looping jadwal
            if belum_dijadwal(1,kontrol)==-1
            break;
            end
        end
    end
    belum_dijadwal=belum_dijadwal;
end
jadwal=anak_sementara;
gen_jadwal=GEN_JADWAL(jadwal);
KRKK_gen_jadwal=[KR KK gen_jadwal];
g = FUNGSI_EVALUASI1(KR, KK, jadwal);
kontrol=0;
while kontrol<50000 & sum(g)>0 %pilih yang maksimum
    kuliah=sum(floor(2*rand(1,jumlah_sks-1)))+1;
    for k=1:1:jumlah_ruang
        g_akhir=zeros(1,jumlah_sks);
        for l=1:1:slot
            if jadwal(k,l)==kuliah
                baris=floor(jumlah_ruang*rand(1))+1;
                kolom=floor(slot*rand(1))+1;
```

```
            jadwal(k,l)=jadwal(baris,kolom);
            jadwal(baris,kolom)=kuliah;
            %berikut adalah fungsi evaluasi
            gen_jadwal=GEN_JADWAL(jadwal);
            KRKK_gen_jadwal=[KR KK gen_jadwal];
            g_akhir = FUNGSI_EVALUASI1(KR, KK, jadwal);
            g_akhir=sum(g_akhir);
            %akhir dari fungsi evaluasi jadwal
            if sum(g_akhir)<= sum(g)
               g=g_akhir;
               break;
            else
               jadwal(baris,kolom)=jadwal(k,l);
               jadwal(k,l)=kuliah;break;
            end
         end
      end
      if sum(g)==sum(g_akhir)
         break;
      end
   end
    kontrol=kontrol+1;
end
kontrol;
anak=jadwal;
fungsi_eval_1=sum(g);
```

```
function [jadwal,fungsi_eval_1,kontrol]=PENCARIAN_LOKAL_RANDOM(KR, KK, slot)
%berikut akan dibuat matrik isialisasi jadwal J ukuran ruangan kali 45
[jumlah_sks, jumlah_ruang]=size(KR);
[jadwal]=INISIAL_RANDOM(KR, KK, slot);
gen_jadwal=GEN_JADWAL(jadwal);
KRKK_gen_jadwal=[KR KK gen_jadwal];
g = FUNGSI_EVALUASI1(KR, KK, jadwal);
kontrol=0;
while kontrol<50000 & sum(g)>0 %pilih yang maksimum
   kuliah=floor(jumlah_sks*rand(1))+1;
   for k=1:1:jumlah_ruang
      g_akhir=zeros(1,jumlah_sks);
      for l=1:1:slot
         if jadwal(k,l)==kuliah
            baris=floor(jumlah_ruang*rand(1))+1;
            kolom=floor(slot*rand(1))+1;
            jadwal(k,l)=jadwal(baris,kolom);
            jadwal(baris,kolom)=kuliah;
            %berikut adalah fungsi evaluasi
            gen_jadwal=GEN_JADWAL(jadwal);
            KRKK_gen_jadwal=[KR KK gen_jadwal];
            g_akhir = FUNGSI_EVALUASI1(KR,KK, jadwal);
            g_akhir =sum(g_akhir);
            %akhir dari fungsi evaluasi jadwal
            if sum(g_akhir)<= sum(g)
               g=g_akhir;
               break;
```

```
            else
               jadwal(baris,kolom)=jadwal(k,l);
               jadwal(k,l)=kuliah;break;
            end
         end
      end
      if sum(g)==sum(g_akhir)
         break;
      end
   end
    kontrol=kontrol+1;
end
fungsi_eval_1=sum(g);
kontrol


function anak=PENCARIAN_LOKAL_SLOT(KRKK, anak_sementara, belum_dijadwal)
%berikut akan dibuat matrik isialisasi jadwal J ukuran ruangan kali 45
[jumlah_sks, jumlah_ruang_plusK]=size(KRKK);
jumlah_ruang=jumlah_ruang_plusK-jumlah_sks;
KR=KRKK(:,1:jumlah_ruang);
KK=KRKK(:,jumlah_ruang+1:jumlah_ruang_plusK);
[satu, banyak]=size(belum_dijadwal);
jadwal=-ones(jumlah_ruang, (24+banyak));
jadwal=anak_sementara;
for kuliah=1:banyak
   if belum_dijadwal(1,banyak)>0
      waktu=24+banyak;
```

```
        jadwal(1,waktu)=belum_dijadwal(1,banyak);
    end
end
[jumlah_ruang, jjk]=size(anak_sementara);
slot=input('masukkan slot :');
[slot,jadwal]=PENYESUAIAAN_SLOT(jadwal, slot);
gen_jadwal=GEN_JADWAL(jadwal);
KRKK_gen_jadwal=[KRKK gen_jadwal];
g=FUNGSI_EVALUASI1(KRKK, gen_jadwal);
kontrol=0;
while kontrol<50000 & sum(g)>0 %pilih yang maksimum
    kuliah=sum(floor(2*rand(1,jumlah_sks-1)))+1;
    for k=1:1:jumlah_ruang
        g_akhir=zeros(1,jumlah_sks);
        for l=1:1:slot
            if jadwal(k,l)==kuliah
                baris=floor(jumlah_ruang*rand(1))+1;
                kolom=floor(slot*rand(1))+1;
                jadwal(k,l)=jadwal(baris,kolom);
                jadwal(baris,kolom)=kuliah;
                %berikut adalah fungsi evaluasi
                gen_jadwal=GEN_JADWAL(jadwal);
                KRKK_gen_jadwal=[KRKK gen_jadwal];
                g_akhir = FUNGSI_EVALUASI1(KRKK,gen_jadwal);
                g_akhir=sum(g_akhir);
                %akhir dari fungsi evaluasi jadwal
                if sum(g_akhir)<= sum(g)
```

```
                g=g_akhir;
                break;
            else
                jadwal(baris,kolom)=jadwal(k,l);
                jadwal(k,l)=kuliah;break;
            end
        end
    end
    if sum(g)==sum(g_akhir)
        break;
    end
  end
  kontrol=kontrol+1;
end
anak=jadwal;
fungsi_eval_1=sum(g);


function [slot,jadwal]=PENYESUAIAAN_SLOT(jadwal,slot)
%fungsi ini menyesuaikan dengan jumlah slot yang diinginkan
[jumlah_ruang,waktu_mak]=size(jadwal);
if waktu_mak > slot % memasukkan kuliah ke waktu yang ada
  for k=1:1:jumlah_ruang
    for l=(slot+1):1:waktu_mak  % pilih kuliah yang masuk pada waktu diluar jadwal
      while jadwal(k,l)>0%pilih waktu dan ruang yang cocok
        for i=jumlah_ruang:-1:1
          for j=slot:-1:1
            if jadwal(i,j)==-1
```

```
                jadwal(i,j)=jadwal(k,l);
                jadwal(k,l)=-1;break;
                end
            end
                if jadwal(k,l)==-1;break;
                end
        end
    end
  end
  jadwal=jadwal(:,1:slot);
else
  jadwal(:,(waktu_mak+1):slot)=-1;
end
```