TELCOMATICS, Vol.1 No.1, November 2016, pp. 1- 11
ISSN : 2541-5867

Perancangan Sistem Penjadwalan Ujian pada suatu Fakultas dengan Beberapa Jurusan dengan Menggunakan Algoritma Genetika

Suwarno
Universitas Internasional Batam
Jl. Gajah Mada, Baloi-Sei Ladi, Batam 29422, (0778) 7437111, Fax. (0778)7437112
e-mail: suwarno.liang@uib.ac.id

Abstract

This research provides an insight into heuristic scheduling by using Genetic Algorithm. Scheduling is a complex process because it must comply with several constraints at once, within a reasonable process. The research uses sample of exam schedule at University of Surabaya, Engineering faculty, with some majors. The results shows that number of chromosome will determine value of objective function obtained greater but longer time taken, and determine the saddle point obtained vanishingly small, it means faster experiencing a stable condition or a condition where the rate of change of small value does not even exist.
Keywords: Heuristic, Genetic algorithm, Objective function

Abstrak

Penelitian ini menunjukkan suatu pengetahuan mengenai penjadwalan heuristic menggunakan algoritma genetika. Penjadwalan adalah suatu proses yang kompleks dikarenakan harus sesuai dengan batasan-batasan yang ada di dalam proses yang jelas. Penelitian ini menggunakan sampel dari penjadwalan ujian di Universitas Surabaya pada Fakultas Teknik. Hasilnya menunjukkan bahwa jumlah kromosom akan menentukan nilai dari fungsi obyektif yang didapatkan walaupun waktu yang dibutuhkan juga lebih lama, dan hal ini juga menentukan kecilnya saddle point yang didapatkan. Ini berarti kondisi stabilnya lebih lama atau suatu kondisi dimana nilai rata-rata perubahan yang kecil, tidak ditemukan.
Keywords: Heuristic, Algoritma Genetik, Fungsi obyektif

I. PENDAHULUAN

Sistem pembelajaran di jenjang pendidikan tinggi menggunakan Sistem Kredit Semester (SKS), namun berbeda implementasinya dengan SKS yang diterapkan pada Sekolah Menengah Atas/Kejuruan. Perbedaannya adalah setiap siswa SMU/K pada tingkat kelas tertentu memiliki mata pelajaran yang telah ditentukan oleh pihak sekolah, sedangkan pada perguruan tinggi, setiap mahasiswa boleh mengambil mata kuliah apapun sesuai dengan mata kuliah yang dibuka. Sistem ini akan menguntungkan pihak mahasiswa, dimana memberi kemungkinan lebih luas kepada mahasiswa untuk memilih program menuju suatu jenjang akademik jurusan tertentu. Hal ini juga berpengaruh dengan masa studi mahasiswa, dimana masa studi setiap mahasiswa dapat berbeda, semakin cakap, dan giat belajar mahasiswa tersebut semakin dapat menyelesaikan studi dalam waktu yang singkat, dan sebaliknya semakin malas, maka mahasiswa tersebut akan semakin lama masa studinya. Mahasiswa yang memiliki masa studi normal berarti ia mengambil mata kuliah sesuai dengan mata kuliah wajib pada tiap semester.

Pihak fakultas dituntut untuk membuat jadwal ujian yang sesuai dengan kebutuhan mahasiswa. Kebutuhan tiap mahasiswa berbeda, sehingga mata kuliah yang dibuka harus disesuaikan dengan kebutuhan mahasiswa. Setelah mahasiswa melakukan perwalian, maka fakultas akan membuat jadwal ujian yang tidak membuat beban belajar mahasiswa semakin berat, waktu belajar lebih lama, tidak ada mahasiswa yang memiliki jadwal ujian yang bentrok, tidak melebihi kapasitas yang disediakan, dan dosen pengajar dapat menjaga ujian untuk

TELCOMATICS, Vol.1 No.1, November 2016, pp. 1-11
ISSN : 2541-5867

mata kuliah yang diasuhnya. Jadwal ujian diatur berdasar *slot* yaitu hari dan jam ujian.

Pembuatan jadwal secara umum terbagi atas dua cara, yaitu: pembuatan jadwal secara manual dan terkomputerisasi. Pembuatan jadwal secara manual mempunyai kelemahan pada kecepatan dan ketelitian. Oleh karena itu dipilih cara terkomputerisasi. Pembuatan jadwal secara terkomputerisasi terbagi lagi menjadi dua cara, yaitu: secara *exhaustic search* yaitu menelusuri semua kemungkinan jadwal yang ada untuk mencari jadwal yang diinginkan dan *heuristic* yaitu pencarian jadwal namun dengan dipandu oleh sebuah aturan. *Exhaustic search* tidak dapat dilakukan walaupun dengan komputer yang sangat cepat karena jumlah jadwal yang ada sangat besar sehingga butuh waktu yang lama untuk menemukan jadwal yang diinginkan. Oleh karena itu cara yang dipilih adalah *heuristic*. *Heuristic* memiliki banyak metode yang bisa digunakan untuk menyusun jadwal seperti algoritma genetika, *taboo search.* Dalam penelitian ini metode yang dipilih adalah algoritma genetika.

Penelitian ini membuat program aplikasi untuk melakukan penjadwalan ujian untuk mencapai kondisi yang diinginkan tanpa melanggar batasan yang diberikan dengan menggunakan algoritma genetika pada Universitas Surabaya, Fakultas Teknik dengan 6 (enam) jurusan yaitu: Teknik Elektro, Teknik Kimia, Teknik Industri, Teknik Informatika dan Teknik Manufaktur pada tahun 1998.

Berikut ini *constraint* atau batasan permasalahan dalam penjadwalan ujian:

1. Mahasiswa boleh mengambil mata kuliah apa pun tanpa perlu harus disesuaikan dengan jadwal ujian yang ada, hal ini diimplementasikan dengan matriks mata kuliah yang boleh atau tidak boleh tabrakan (Gambar 1). Matriks ini merupakan matriks yang simetris terhadap diagonal utama.

MK \ MK	A	B	C	D
A	√	X	√	√
B	X	√	√	√
C	√	√	√	√
D	√	√	√	√

√ boleh tabrakan
X tidak boleh tabrakan

Gambar 1. Matriks Tabrakan

2. Pengajar yang sama untuk mata kuliah yang berbeda tidak boleh diujikan pada slot yang sama. Tabel 1 menunjukkan salah satu kemungkinan di mana mata kuliah A dan C tidak boleh diujikan pada slot yang sama karena diajar oleh dosen yang sama.
3. Jumlah total mahasiswa pada sebuah slot tidak boleh melebihi kapasitas kursi yang disediakan.
4. Jadwal ujian tidak boleh diadakan pada slot dimana dosen pengajar berhalangan.

Tabel 1. Pengajar Mata Kuliah

Mata Kuliah	Dosen
A	I
B	II
C	I
D	III

Berikut ini *preference* atau kondisi yang diinginkan namun boleh tidak dipenuhi, meliputi*:*

1. Jadwal untuk tiap jurusan seimbang, tidak boleh ada jurusan tertentu yang memonopoli jadwal ujian.
2. Mata kuliah pada semester yang sama untuk satu jurusan tidak boleh diletakkan pada hari yang sama
3. Mata kuliah pada semester yang selisih dua untuk satu jurusan tidak boleh diletakkan pada hari yang sama
4. Ujian untuk tiap mata kuliah pada semester yang sama diharapkan untuk diadakan dengan jadwal yang dipisahkan dengan libur (Tabel 2).

Tabel 2. Hari Ujian Mata Kuliah dengan Jurusan dan Semester Sama

A	Libur	B	Libur	C	Libur	D

Jadi dengan 4 buah *constraint* dan 4 buah *preference* di atas maka kapasitas kursi dihitung berdasar jumlah total kursi yang disediakan pada tiap slot, mata kuliah yang diujikan merupakan mata kuliah yang telah diatur berdasar jurusan dan semester, dan hanya mencakup hari dan jam ujian Penjadwalan ini tidak mencakup mata kuliah

yang diujikan pada beberapa fakultas, mata kuliah pilihan, ruang dan dosen pengawas.

Tujuan yang ingin dicapai dari penelitian ini adalah:

1. Membuat program dengan metode *heuristic* yaitu algoritma genetika.
2. Menentukan parameter optimal sesuai dengan algoritma genetika berdasarkan beberapa contoh *data set*.

Manfaat program penjadwalan bagi pengguna program sebagai berikut:

1. Penjadwalan ujian dapat dilakukan dalam waktu yang relatif lebih cepat, dan lebih teliti daripada dengan cara manual maupun metode *exhaustic*.
2. Program penjadwalan ujian akan menghasilkan sebuah jadwal yang optimal sesuai dengan algoritma genetika

Di samping itu juga akan memberikan manfaat bagi mahasiswa yaitu:

1. Beban belajar mahasiswa lebih ringan.
2. Waktu belajar mahasiswa lebih panjang.
3. Ujian mata kuliah yang diambil mahasiswa tidak bentrok.
4. Dosen dapat mendampingi mahasiswa pada saat mata kuliah yang diampu diujikan.

II. KAJIAN PUSTAKA

A. Heuristic

Masalah penjadwalan merupakan masalah *Non Linear Programming-complete (NP-complete),* sebab pembuatan jadwal bersifat sangat kompleks. Banyak ilmuwan ilmu komputer berpendapat bahwa permasalahan *NP-complete* sulit dipecahkan, jika sebuah permasalahan NP-complete dapat dipecahkan dalam waktu *polynomial*, maka setiap permasalahan NP-complete akan dapat diselesaikan dalam waktu *polynomial* juga, karena semua permasalahan *NP-complete* memiliki kompleksitas waktu yang sama. Sampai saat ini, belum ada algoritma untuk waktu *polynomial* yang ditemukan untuk menyelesaikan permasalahan *NP-complete*. Untuk memecahkan permasalahan *NP-complete*, metode *heuristic* dapat digunakan untuk memecahkan permasalahan ini.

Heuristic adalah bagian dari metode kecerdasan buatan yang dapat digunakan untuk menemukan solusi dalam waktu yang layak untuk dilakukan tetapi tidak menjamin ditemukannya solusi optimal. Sehingga perbedaan utama dari algoritma dan *heuristic* adalah jaminan ditemukannya solusi optimal. Perbedaan lainnya adalah *heuristic* bergantung pada *rules of thumb* yaitu aturan-aturan yang diperoleh dari intuisi, pertimbangan, pengalaman dan sebagainya selain metode yang menggunakan analisis. Walaupun *heuristic* tidak memberikan jaminan pada ditemukannya solusi optimal, metode ini tetap banyak digunakan. *Heuristic* dipilih karena lebih baik memperoleh solusi yang kualitasnya cukup baik dalam waktu yang wajar daripada memperoleh solusi terbaik dalam waktu berabad-abad.

Untuk memberikan alasan mengapa dipilihnya metode *heuristic* dibandingkan cara *exhaustic search*. Program ilustrasi ini menyusun jadwal dengan cara langsung membuat suatu jadwal secara acak lalu menguji jadwal tersebut apakah melanggar *constraint* dan menghitung nilai total dari semua *preference*. Istilah slot yang digunakan berikut ini berarti jam ujian tertentu pada hari tertentu. Misalkan terdapat 184 mata kuliah dan 48 slot maka banyak jadwal yang bisa disusun adalah 184! / (184 - 48)! + 136! / (136 - 48)! + 88! / (88 – 48)! + 40! = 6.09×10^{105} jadwal. Program ini mampu menghasilkan sebuah jadwal sekaligus menguji *constraint* dan menghitung nilai *preference* dalam 10^{-12} detik. Berarti total waktu yang diperlukan oleh program ini untuk menghasilkan jadwal yang diinginkan adalah $6.09 \times 10^{105} \times 10^{-12}$ detik atau 6.09×10^{93} detik atau 1.93×10^{84} abad. Dari ilustrasi ini secara sekilas terlihat pembuatan jadwal dengan cara *exhaustic search* tidak memadai untuk diterapkan.

B. Algoritma Genetika

Seperti yang diungkapkan oleh David E. Goldberg [2] bahwa algoritma genetika merupakan algoritma pencarian yang berdasar pada mekanisme *natural* selection dan *natural genetic*. Dari *natural selection* diambil informasi berupa *bit string* dihasilkan *bit string* baru yang merupakan hasil dari *natural genetic*.

Terdapat 4 langkah yang ditempuh dalam algoritma genetika antara lain: (1) pengkodean berupa *binary representation* atau *non binary representation,* (2) membuat populasi awal dengan n kromosom, (3)

TELCOMATICS, Vol.1 No.1, November 2016, pp. 1-11
ISSN : 2541-5867

menggunakan *objective function* sebagai tujuan yang ingin dicapai, dan (4) menggunakan *probabilistic transition rules* bukan *deterministic rules*. Secara umum terdapat 3 (tiga) operator penting dalam algoritma genetika antara lain: reproduksi, *crossover*, dan mutasi.

Penerapan algoritma genetika pada penjadwalan ujian ini, pengkodean yang dilakukan berupa *non-binary representation*, berupa *table representation*, pengkodean berupa tabel.

Langkah-langkah yang dilakukan dalam algoritma genetika sebagai berikut:

1. Inisialisasi awal adalah menciptakan sebuah populasi yang terdiri atas n kromosom. Kromosom berupa jadwal ujian. Tiap jadwal diurutkan berdasarkan kode mata kuliah.
2. Cari sebuah jadwal yang masih melanggar *constraint* atau belum memenuhi ketentuan *preference* dan tentukan mata kuliah mana yang akan diubah hari dan jam ujiannya. Posisi mata kuliah ini disebut sebagai titik potong.
3. Pengambilan nilai probabilitas secara acak untuk menentukan apakah jadwal tersebut akan dilakukan proses *crossover* atau mutasi.
4. Jika proses *crossover* yang dilakukan maka cari pasangan jadwal lain dengan titik potong sama. *Record* yang ditunjuk oleh titik potong tersebut pada kedua jadwal yang terpilih disebut sebagai pasangan *allele* pada dua buah jadwal. Jika proses mutation yang dilakukan maka tentukan pasangan *allele* dalam sebuah jadwal dan lakukan penukaran slot. *Allele* merupakan sebuah record pada suatu jadwal yang berisi kode mata kuliah, semester, jumlah mahasiswa, hari dan jam ujian.
5. Reproduksi: mengambil sejumlah jadwal yang memiliki nilai *objective function* terbaik dari $populasi_i$ dan $populasi_{i-1}$ sebanyak ½ n; dan simpan sebagai $populasi_{i-1}$. Lakukan penyalinan $populasi_{i-1}$ sebagai $populasi_i$ dan lakukan proses *crossover* atau mutasi terhadap $populasi_i$ berdasar nilai probabilitas yang ditentukan secara acak. Karena $populasi_i$ masih berjumlah setengah kali jumlah populasi awal maka harus dibuat sejumlah jadwal yang baru sebanyak ½ n secara acak sehingga $populasi_i$ berjumlah n jadwal.
6. Lakukan proses ke-2 sampai proses ke-5 hingga mencapai kondisi berakhir. Kondisi berakhir ditentukan berdasar jika memenuhi batas iterasi atau sudah tidak memiliki perubahan nilai selama beberapa kali iterasi. Dari tiap iterasi diambil sebuah jadwal dengan nilai *objective function* terbaik dari satu generasi.

III. METODE

A. Analisis Sistem

Jadwal ujian dikategorikan sebagai jadwal yang baik apabila dapat menguntungkan mahasiswa sebanyak mungkin. Sangat banyak faktor yang mempengaruhi suatu jadwal yang menguntungkan mahasiswa peserta ujian. Berikut ini beberapa faktor-faktor yang dapat menguntungkan mahasiswa:

1. Dosen hadir pada saat mata kuliah yang diampu diujikan. Alasan dari hal ini adalah jika terdapat permasalahan dengan soal ujian dapat langsung ditangani oleh dosen yang bersangkutan. Hal lainnya adalah kehadiran dosen pengasuh dapat membantu siswa secara psikologis dalam menghadapi ujian.
2. Dosen pengajar yang hadir saat ujian tidak boleh terpecah konsentrasinya dengan menjaga lebih dari satu mata kuliah pada suatu slot.
3. Sejumlah mata kuliah dapat diatur untuk tidak terletak dalam satu slot. Hal ini menyebabkan jadwal menjadi fleksibel atau dapat diatur sesuai dengan keinginan mahasiswa.
4. Mata kuliah yang berada pada suatu slot tidak boleh melebihi kapasitasnya. Hal ini mutlak harus dilakukan karena jika peserta ujian melebihi kapasitas kursi akan menyebabkan kekacauan pada saat ujian dilaksanakan. Kekacauan ini pada akhirnya akan merusak konsentrasi peserta ujian.
5. Mata kuliah dengan jurusan dan semester sama tidak pada satu hari karena kebanyakan mahasiswa

mengambil mata kuliah dengan jurusan dan semester yang sama. Mahasiswa mengikuti ujian sekali dalam sehari sehingga beban belajar mahasiswa merasakan tidak terlalu berat.

6. Mata kuliah dengan jurusan dan semester sama mendapat selang libur. Hal ini dimaksudkan agar mahasiswa bisa belajar lebih optimal.
7. Mata kuliah dengan jurusan sama dan semester selisih dua tidak sehari supaya mahasiswa yang terlambat dan terlalu cepat mengambil mata kuliah tidak mengalami ujian lebih dari sekali dalam sehari.
8. Jumlah mahasiswa suatu jurusan yang mengikuti ujian pada suatu hari tidak melebihi porsinya. Hal ini dilakukan agar tidak timbul kecemburuan terhadap suatu jurusan yang dominan pada hari tertentu.

B. Desain dan Implementasi Sistem

Untuk program pembuatan jadwal ini diperlukan data untuk diolah, maka diperlukan sebuah basis data untuk ditulis, dibaca dan diubah. Dengan adanya basis data ini pemakai dapat dengan mudah memanipulasi data yang akan dipergunakan, input data yang dimasukkan berbeda-beda sesuai keinginan pemakai.

Rancangan basis data

Table 1: Dosen

Table 1 digunakan untuk menyimpan data semua dosen sebuah fakultas.

Table 2: Mata Kuliah

Table 2 digunakan untuk menyimpan data semua mata kuliah yang dimiliki oleh sebuah fakultas.

Table 3: Kapasitas

Table 3 digunakan untuk menyimpan data banyak kursi mahasiswa yang bisa ditampung untuk tiap hari pada jam ujian tertentu.

Table 4: Pengajar Mata Kuliah

Table 4 digunakan untuk menyimpan data pengajar mata kuliah pada kelas paralel tertentu. Tiap kelas paralel dapat diajar oleh satu atau beberapa dosen.

Table 5: Dosen Absen

Table 5 digunakan untuk menyimpan data pada hari dan jam ke berapa dosen pengajar berhalangan untuk menjaga ujian.

Table 6: Jumlah Peserta Mata Kuliah

Table 6 digunakan untuk menyimpan data jumlah mahasiswa untuk kode mata kuliah tertentu.

Table 7: Mata Kuliah yang Bertabrakan

Table 7 digunakan untuk menyimpan data dua mata kuliah yang bertabrakan dengan jumlah mahasiswa terkait dengan kedua mata kuliah tersebut.

Model Penjadwalan Dengan Algoritma Genetika

Pemeriksaan *Constraint*

Constraint adalah syarat yang tidak boleh dilanggar dalam pembuatan jadwal. Masing-masing *constraint* memiliki cara pengecekan yang berbeda. Pemeriksaan *constraint* dilakukan guna mengetahui apakah perubahan jadwal untuk mata kuliah tertentu boleh dilakukan atau tidak, dan juga untuk mengetahui apakah ada mata kuliah yang melanggar *constraint*. Ada 3 macam *constraint*:

Constraint 1: Bertabrakan

Pemeriksaan *constraint* 1 adalah *constraint* tabrakan, yaitu untuk memeriksa apakah ada mata kuliah terletak pada slot yang sama dengan pasangannya.

Constraint 2: Melebihi Kapasitas

Pemeriksaan *constraint* 2 adalah *constraint* kapasitas pada slot yang disediakan, yaitu untuk memeriksa apakah ada sebuah slot yang digunakan oleh mata kuliah yang terletak pada slot itu hingga melebihi kapasitasnya.

Constraint 3: Diujikan pada Slot Yang Dilarang

Pemeriksaan *constraint* 3 adalah *constraint* mata kuliah diujikan pada hari dan jam yang tidak diperbolehkan karena dosen berhalangan menjaga pada hari dan jam tersebut.

Preference

Preference sebagai syarat yang boleh dilanggar harus mempunyai mekanisme untuk

TELCOMATICS, Vol.1 No.1, November 2016, pp. 1-11
ISSN : 2541-5867

menunjukkan berapa banyak syarat ini dipenuhi atau dilanggar. Untuk itu tiap *preference* harus mempunyai fungsi untuk mengembalikan nilai *preference* itu pada suatu jadwal. Gabungan dari semua nilai *preference* merupakan *objective function*. Ada 4 macam *preference* sebagai berikut:

***Preference* 1: Jurusan Berimbang**

Nilai *preference* jurusan seimbang ditentukan berdasar jumlah peserta mata kuliah jurusan tertentu pada hari tertentu dibandingkan dengan jumlah mahasiswa ideal pada hari tertentu sesuai dengan jurusannya. Jumlah mahasiswa ideal diperoleh berdasar rasio. Jika melebihi jumlah mahasiswa ideal berarti dianggap belum memenuhi *preference* jurusan seimbang, dan akan diberikan nilai semakin banyak yang melebihi jumlah ideal maka semakin kecil nilai *preference* yang dihasilkan.

***Preference* 2: Ujian Jurusan Dan Semester Sama, Sekali Dalam Sehari**

Nilai *preference* ujian jurusan dan semester sama, sekali dalam sehari dihitung berdasar pada hari tertentu, untuk suatu jurusan ujian berapa kali. Jika setiap hari satu ujian maka nilai *preference* yang diperoleh semakin tinggi, sebaliknya jika dalam satu hari terdapat beberapa ujian maka nilai semakin rendah.

***Preference* 3: Ujian Jurusan dan Semester Sama, Mendapatkan Selang Libur**

Nilai *preference* selang libur ditentukan banyaknya pasangan mata kuliah dengan jurusan dan semester sama yang memiliki selisih hari sama lebih besar satu. Jika semakin banyak maka nilai semakin tinggi.

***Preference* 4: Ujian Jurusan Sama dan Semester Selisih Dua, Tidak Sehari**

Penilaian *preference* ini akan memberikan nilai yang lebih baik jika menemukan pasangan mata kuliah dengan jurusan sama dan semester selisih dua dan tidak terletak pada satu hari.

Setelah menentukan *constraint* dan *preference*, dilanjutkan langkah-langkah implementasi sebagai berikut:

Membuat Populasi Awal

Populasi awal berupa sekumpulan jadwal dengan 2 sifat apakah sudah tidak melanggar *constraint* atau melanggar *constraint*.

Mengumpulkan Mata Kuliah yang Melanggar Constraint

Populasi awal masih melanggar *constraint* maka untuk menentukan kromosom mana yang hendak dilakukan *crossover* atau *mutation*, akan dilakukan pengecekan dari setiap kromosom apakah ada yang melanggar *constraint*, dan akan dikumpulkan menjadi sebuah daftar kromosom yang melanggar.

Mengumpulkan Mata Kuliah yang Tidak Memenuhi *Preference*

Jika populasi awal sudah tidak melanggar *constraint*, maka yang harus dilakukan adalah mencari kromosom atau jadwal yang akan diproses berdasar ketentuan *preference* yang ada. Untuk mengetahui yang tidak memenuhi *preference*, dilakukan 4 macam pengecekan *preference* sebagai berikut:

Cek *Preference* 1: Ujian Semester Dan Jurusan Sama, Sekali Dalam Sehari

Pengecekan *preference* ini akan menghasilkan daftar mata kuliah semua jurusan pada sebuah jadwal, yang hari ujiannya sama dengan mata kuliah lainnya yang satu semester dan jurusan.

Cek *Preference* 2: Ujian Jurusan Dan Semester Sama, Mendapatkan Selang Libur

Pengecekan *preference* ini akan menghasilkan daftar mata kuliah yang ujiannya tidak memiliki selang libur dari mata kuliah yang lainnya dengan syarat pada semester dan jurusan sama.

Cek *Preference* 3: Ujian Jurusan Sama Dan Semester Selisih Dua, Tidak Sehari

Pengecekan *preference* ini dilakukan dengan cara mengumpulkan mata kuliah yang hari ujiannya sama, namun memiliki selisih semester sama dengan 2 (dua) pada jurusan yang sama.

Cek *Preference* 4: Jurusan Berimbang

Pengecekan *preference* jurusan seimbang akan menghasilkan kumpulan mata kuliah yang menyebabkan jadwal ujian untuk tiap jurusan tidak seimbang. Jurusan seimbang berarti tidak ada jurusan yang memonopoli dalam satu hari. Penilaian ini berdasar jumlah mahasiswa tiap jurusan dibanding dengan jumlah kursi yang disediakan.

Menukar Hari dan Jam Ujian

Dalam proses *crossover* dan *mutation*, yang dilakukan adalah penukaran hari dan jam ujian pasangan *allele*.

Probabilitas Random

Bilangan random diambil sebanyak 4 digit, guna menentukan proses *crossover* atau mutation.

Titik Potong

Titik potong pada sebuah jadwal dapat dicari pada kumpulan mata kuliah pada sebuah jadwal dan berdasarkan kode mata kuliah dapat ditentukan titik potong pada kromosom pertama.

Kondisi Berakhir

Algoritma genetika akan berhenti jika memenuhi salah satu dari beberapa kondisi, antara lain: mencapai batas maksimum iterasi, selisih nilai *objective function* antar kromosom dalam sebuah populasi sama, atau tidak adanya peningkatan nilai sebanyak n kali iterasi.

Crossover

Proses *crossover* merupakan proses penukaran dua buah slot pada dua buah jadwal. Penukaran melibatkan dua buah jadwal, sehingga memberikan pengaruh perubahan nilai terhadap kedua jadwal tersebut.

Mutasi

Proses mutasi akan melakukan penukaran dua buah slot pada sebuah jadwal, maka perubahan nilai *objective function* hanya terjadi pada jadwal tersebut.

Natural Genetic

Dalam satu generasi atau iterasi, semua kromosom atau jadwal akan mengalami satu kali *crossover* atau satu kali *mutation*.

Natural Selection

Natural selection atau seleksi alam, merupakan proses pengambilan jadwal yang memiliki nilai *objective function* tertinggi sebanyak setengah kali jumlah populasi awal. Pengambilan nilai *objective function* tertinggi dilakukan berdasar populasi yang aktif, dan populasi lama terbaik yang berjumlah setengah kali jumlah populasi awal. Populasi lama ini sudah mewakili populasi-populasi sebelumnya yang memiliki nilai *objective function* tertinggi. Sehingga akan berpengaruh terhadap proses *natural selection*, yaitu yang terbaik yang akan bertahan.

Proses Genetika

Proses genetika merupakan proses utama yang dilakukan dalam mengolah sejumlah jadwal dalam sebuah populasi hingga mencapai kondisi akhir.

IV. HASIL DAN PEMBAHASAN

C. Analisis Parameter Optimal Program

Analisis parameter ini didasarkan pada nilai *objective function* yang diperoleh pada tabel 3. Parameter program dengan algoritma genetika meliputi jumlah kromosom, dan probabilitas *crossover*. Analisis parameter jumlah kromosom dilakukan dengan mengambil jumlah kromosom = 2^n, n = 2,3,4 atau jumlah kromosom = 4,8,16. Sedangkan analisis parameter probabilitas *crossover* dengan mengambil nilai probabilitas *crossover* mulai dari 0.1, 0.2, 0.3, .. 0.9. Nilai *objective function* pada tabel 3 merupakan nilai rata-rata dari 30 nilai *objective function* untuk tiap iterasi. Jumlah iterasi adalah 80. Tabel 3 ini berisikan nilai *objective function* dengan mengambil nilai awal (nilai pada iterasi ke-0) dan akhir (nilai pada iterasi ke-80).

TELCOMATICS, Vol.1 No.1, November 2016, pp. 1-11
ISSN : 2541-5867

Tabel 1. Nilai Objective Function Tertinggi

Jumlah Kromosom / Probabilitas Crossover	4		8		16	
	Nilai Awal	Nilai Akhir	Nilai Awal	Nilai Akhir	Nilai Awal	Nilai Ak
0.1	2605	2717	2615	2738	2632	275
0.2	2605	2728	2615	2748	2632	275
0.3	2605	2733	2615	2756	2632	276
0.4	2605	2738	2615	2754	2632	276
0.5	2605	2742	2615	2760	2632	277
0.6	2605	2740	2615	2766	2632	277
0.7	2605	2747	2615	2769	2632	277
0.8	2605	2754	2615	2772	2632	277
0.9	2605	2756	2615	2770	2632	277
TOTAL	**23445**	**24655**	**23535**	**24833**	**23688**	**249**
RATA-RATA	**2605**	**2739**	**2615**	**2759**	**2632**	**276**
MAKSIMAL	**2605**	**2756**	**2615**	**2772**	**2632**	**277**

Semakin banyak jumlah kromosom, nilai rata-rata dan nilai maksimal yang dicapai semakin meningkat. Peningkatan nilai *objective function* maksimal awal yaitu dari 2605 menjadi 2615 adalah 10 poin, dan dari 2615 menjadi 2632 adalah 17 poin. Peningkatan nilai *objective function* maksimal akhir yaitu dari 2756 menjadi 2772 adalah 16 poin, dan dari 2772 menjadi 2778 adalah 6 poin.

Semakin banyak jumlah kromosom nilai awal semakin meningkat, dan semakin banyak jumlah kromosom nilai akhir juga mengalami peningkatan, namun peningkatan nilai akhir akan semakin rendah yaitu dari 16 poin menjadi 6 poin. Jumlah kromosom juga berpengaruh terhadap waktu yang ditempuh, semakin banyak jumlah kromosom waktu yang ditempuh semakin banyak. Untuk kasus ini, penambahan waktu yang ditempuh adalah dua kali. Jadi jumlah kromosom makin banyak maka nilai *objective function* maksimal akhir yang diperoleh semakin tinggi, namun waktu yang ditempuh makin lama. Dari tabel 3 dapat diketahui bahwa semakin tinggi nilai probabilitas *crossover* menunjukkan semakin meningkat nilai akhir yang diperoleh.

Tabel 4 berisikan *saddle point* yang diperoleh dengan menggunakan populasi awal tanpa melanggar *constraint*. Dari tabel 4 dapat diketahui perkiraan pada iterasi ke berapa sebuah grafik akan mengalami perubahan. Perkiraan ini diperoleh dari nilai rata-rata *saddle point* pada tiap jumlah kromosom dengan nilai probabilitas dari 0.1 hingga 0.9. Populasi dengan jumlah kromosom=4 akan mencapai *saddle point* pada iterasi ke-48, populasi dengan jumlah kromosom=8 akan mencapai *saddle point* pada iterasi ke-40, populasi dengan jumlah kromosom=16 akan mencapai *saddle point* pada iterasi ke-36. Populasi dengan jumlah kromosom=16 memiliki nilai rata-rata *saddle point* paling kecil, maka dapat disimpulkan semakin banyak jumlah kromosom maka *saddle point* semakin kecil, berarti grafik semakin cepat mencapai kondisi mendatar atau tingkat perubahan nilai *objective function* kecil bahkan tidak ada.

Tabel 2. Saddle Point

Jumlah Kromosom / Probabilitas Crossover	4	8	16
0.1	38	29	24
0.2	29	29	25
0.3	54	29	40
0.4	39	44	32
0.5	48	51	31
0.6	46	37	44
0.7	61	40	44
0.8	67	41	51
0.9	47	60	35
TOTAL	**429**	**360**	**326**
RATA-RATA	**48**	**40**	**36**
MAKSIMAL	**67**	**60**	**51**

Dari hasil analisis tabel 3 dan tabel 4 maka dapat disimpulkan parameter optimal adalah jumlah kromosom=8, dan probabilitas *crossover*=0.9, karena dilihat dari segi waktu, populasi dengan jumlah kromosom 8 akan mencapai nilai akhir dengan waktu setengah kali dari populasi dengan jumlah kromosom 16 meskipun populasi dengan 16 kromosom ini memiliki nilai akhir yang lebih tinggi, sebab kenaikan waktu sebanding dengan kenaikan jumlah kromosom, yaitu jumlah kromosom berjumlah 2 kali dari sebelumnya maka didapat waktu yang ditempuh juga 2 kali dari sebelumnya.

D. Evaluasi Hasil

Jadwal yang dihasilkan dievaluasi dari dua segi yaitu: *contraint* dan *preference*.

Evaluasi pada *Constraint*

Pada pembuatan jadwal ini jadwal tidak pernah melanggar *constraint* kecuali populasi awal sudah melanggar *constraint*, namun jadwal dengan nilai *objective function* tertinggi bisa tidak melanggar *constraint* dengan syarat telah diulang sebanyak x kali generasi. Adanya hasil jadwal yang masih melanggar *constraint* dapat disebabkan pula oleh banyaknya batasan yang harus dipatuhi.

Evaluasi pada *Preference*

Di bawah ini akan diberikan evaluasi hasil dari algoritma genetika melakukan optimasi meningkatkan nilai dari tiap *preference*. Untuk evaluasi ini akan menggunakan 2 jenis populasi yaitu populasi dengan 16 kromosom dan probabilitas *crossover* =0.9, dan populasi dengan 32 kromosom dan probabilitas *crossover*=0.4. Pengambilan contoh dengan dua populasi di atas berdasarkan hasil Tabel 3 di mana diambil dua jadwal tertinggi, yang masing-masing merupakan anggota dua populasi yang berbeda seperti yang telah ditulis di atas. Kedua jadwal ini dibandingkan dengan jadwal yang dihasilkan pada saat populasi awal. Berikut daftar nilai *preference* untuk jadwal dengan nilai *objective function* tertinggi pada populasi 16 kromosom, dan probabilitas *crossover*=0.9.

Tabel 3. Evaluasi Preference

Jenis *Preference*	Populasi Awal	Populasi ke-1024
Jurusan Berimbang	769	771
Jurusan dan Semester Sama diujikan pada hari yang berbeda	805	905
Jurusan dan Semester Sama diujikan dengan selang libur	389	440
Jurusan Sama dan Semester Selisih Dua diujikan pada hari yang berbeda	679	700
T o t a l	2642	2816

Tabel 5 berisi daftar nilai *preference* untuk jadwal dengan nilai *objective function* tertinggi pada populasi 32 kromosom, dan probabilitas *crossover*=0.4.

Jurusan Berimbang

Untuk mengevaluasi optimasi *preference* ini digunakan dua buah pasangan nilai awal dan nilai setelah optimasi yaitu: (769,774) dan (769,771). Dari selisih kedua pasangan yaitu 5 (lima) dan 2 (dua), terlihat tidak ada peningkatan kualitas yang berarti. Hal ini bisa terjadi karena selama proses *natural genetic* terjadi penukaran dua buah slot yang menyebabkan penilaian terhadap *preference* jurusan seimbang berubah.

Ujian Jurusan dan Semester Sama, Sekali Dalam Sehari

Untuk mengevaluasi optimasi *preference* ini digunakan dua buah pasangan nilai awal dan nilai setelah optimasi yaitu: (805,905) dan (805,905). Selisih antara kedua pasangan nilai sama yaitu 100 (seratus), maka dapat disimpulkan nilai optimal untuk *preference* ini dicapai pada nilai 905, walaupun ada kemungkinan meningkat namun harus dilakukan hingga generasi berikutnya, tetapi tidak menutup kemungkinan nilai *preference* ini menurun tetapi nilai *preference* yang lain meningkat.

Ujian Jurusan dan Semester Sama, Mendapatkan Selang Libur

Untuk mengevaluasi optimasi *preference* ini digunakan dua buah pasangan nilai awal dan nilai setelah optimasi yaitu: (389,440) dan (389,440). Selisih antara kedua pasangan nilai sama yaitu 51 (lima puluh satu), maka dapat disimpulkan nilai optimal untuk *preference* ini dicapai pada nilai 440, walaupun ada kemungkinan meningkat namun harus dilakukan hingga generasi berikutnya, tetapi tidak menutup kemungkinan nilai *preference* ini menurun tetapi nilai *preference* yang lain meningkat.

Ujian Jurusan Sama Dan Semester Selisih Dua, Tidak Sehari

Untuk mengevaluasi optimasi *preference* ini digunakan dua buah pasangan nilai awal dan nilai setelah optimasi yaitu: (679,696) dan (679,700). Dari selisih kedua pasangan terlihat peningkatan 17 dan 18 mata kuliah dengan jurusan sama dan semester selisih dua tidak sehari.

E. Evaluasi Perbandingan Jadwal Manual Dengan Jadwal Hasil Optimasi Menggunakan Algoritma Genetika

Jadwal manual merupakan jadwal ujian yang dibuat tanpa menggunakan bantuan komputer yaitu Jadwal UTS-UAS Genap Jurusan Teknik Informatika. Sebelumnya diasumsikan mata kuliah pilihan diletakkan pada semester 9. Sebagai bahan perbandingan adalah jadwal hasil optimasi dengan algoritma genetika yang memiliki parameter jumlah kromosom=32, probabilitas *crossover*=0.4,

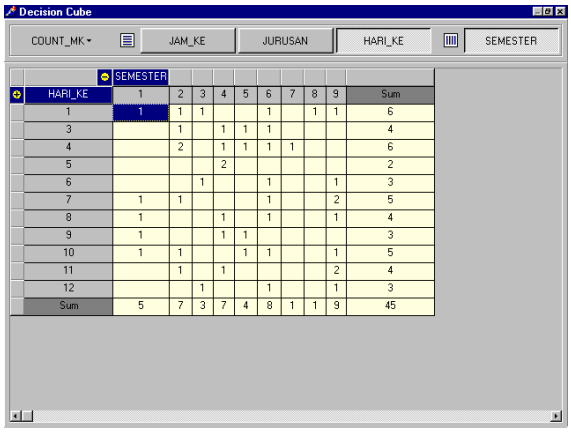
TELCOMATICS, Vol.1 No.1, November 2016, pp. 1-11
ISSN : 2541-5867

dan dilakukan iterasi hingga 100 kali generasi. Populasi awal diciptakan tanpa melanggar *constraint*. Nilai *objective function* tertinggi = 477 (Tabel 6).

Tabel 4. Nilai *Objective Function* Jadwal Manual

Jenis Preference	Nilai
Jurusan Berimbang	0
Jurusan dan Semester Sama diujikan pada hari yang berbeda	225
Jurusan dan Semester Sama diujikan dengan selang libur	108
Jurusan Sama dan Semester Selisih Dua diujikan pada hari yang berbeda	144
T O T A L	477

Nilai *preference* jurusan seimbang=0 dikarenakan jurusan hanya ada satu. Untuk Untuk menganalisis jadwal manual digunakan decision cube seperti pada Gambar 2.

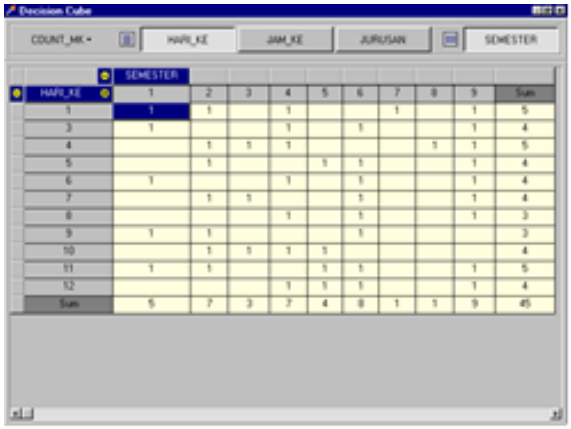


Gambar 2. *Decision Cube* Jadwal Manual

Pada jadwal manual akan dianalisis berdasar tiap *preference* sebagai berikut:

1. Jurusan Berimbang
   Untuk *preference* tidak dilakukan sebab hanya ada satu jurusan.
2. Ujian Jurusan dan Semester Sama, Sekali Dalam Sehari
   Pada jadwal manual masih ditemukan mata kuliah dengan semester sama diujikan pada hari yang sama yaitu pada hari ke-4,5,7.
3. Ujian Jurusan dan Semester Sama, Mendapatkan Selang Libur
   Pada jadwal manual nampak sekali ujian dilaksanakan tanpa selang libur dialami mahasiswa semester 1,2,4,5,6,8,9, salah satu contohnya mata kuliah semester ke-2 dilaksanakan ujian pada hari ke-3,4,10,11. Hal ini dapat menyebabkan mahasiswa akan belajar setiap hari, dan tidak memiliki selang libur.
4. Ujian Jurusan Sama dan Semester Selisih Dua, Tidak Boleh Sehari.
   Pada jadwal manual masih ditemukan 8 (delapan) pasangan mata kuliah selisih dua yang ujiannya dilakukan pada satu hari yang sama, salah satu contohnya: hari ke-3 terdapat 2 pasangan mata kuliah yaitu mata kuliah semester 2 & 4, 4 & 6. Mahasiswa yang mengambil mata kuliah semester 2 dan 4 sekaligus akan mengalami beban belajar yang lebih banyak sebab terletak pada hari yang sama. Kasus ini sering dialami oleh mahasiswa yang mengambil terlalu cepat atau terlalu lambat.

Untuk menganalisis jadwal hasil optimasi algoritma genetika digunakan decision cube.



Gambar 3. *Decision Cube* Jadwal Hasil Optimasi

Pada jadwal hasil optimasi menggunakan algoritma genetika akan dianalisis berdasar tiap *preference* sebagai berikut:

1. Jurusan Berimbang
   Untuk *preference* tidak dilakukan sebab hanya ada satu jurusan.
2. Ujian Jurusan dan Semester Sama, Sekali Dalam Sehari
   Preference kedua ini terpenuhi dengan baik, dimana tiap mata kuliah pada tiap semester diujikan satu hari satu kali, tidak ada yang sampai lebih dari satu mata kuliah dalam satu hari. Jika dibandingkan dengan jadwal manual maka jadwal hasil optimasi lebih unggul dalam memenuhi *preference* kedua ini.

Nilai *preference* yang telah dicapai yaitu nilai 225 merupakan nilai *preference* yang sudah optimal.

3. Ujian Jurusan dan Semester Sama, Mendapatkan Selang Libur
   Pada jadwal optimasi ujian dilaksanakan tanpa selang libur dialami mahasiswa semester 2,4,6,9 saja.
4. Ujian Jurusan Sama dan Semester Selisih Dua, Tidak Sehari.
   Pada jadwal hasil optimasi terdapat 9 pasang semester, berarti dalam memenuhi *preference* keempat, jadwal hasil optimasi belum bisa mencapai target.

Berdasarkan hasil evaluasi kedua jenis jadwal di atas dapat disimpulkan bahwa jadwal hasil optimasi lebih baik daripada jadwal manual, disamping dari segi waktu yang lebih cepat, juga dalam hal pencapaian target tiap *preference*.

V. KESIMPULAN

Dari penelitian ini, dapat diambil kesimpulan sebagai berikut:

1. Sesuai dengan tujuan dari penelitian ini adalah untuk pengembangan ilmu *heuristic*, dimana program ini berusaha mengaplikasikan konsep-konsep yang ada pada *heuristic*, khususnya konsep pemanduan pencarian lokal dengan algoritma genetika. Program penjadwalan dengan algoritma genetika dapat menghasilkan jadwal yang berkualitas dalam waktu yang relatif tidak terlalu lama jika dibandingkan dengan cara manual.
2. Algoritma genetika dapat digunakan untuk menghasilkan jadwal sesuai dengan batasan yang diberikan.
3. Parameter jumlah kromosom yang semakin banyak maka nilai *objective function* yang diperoleh makin besar namun waktu yang ditempuh makin lama.
4. Parameter jumlah kromosom juga menentukan *saddle point*, dimana semakin banyak jumlah kromosom maka *saddle point* yang diperoleh makin kecil, berarti semakin cepat mengalami kondisi yang stabil atau kondisi dimana tingkat perubahan nilai kecil bahkan tidak ada.
5. Parameter probabilitas *crossover* yang semakin besar maka nilai *objective function* yang diperoleh makin besar, dan waktu yang ditempuh rata-rata sama. Nilai probabilitas *crossover* semakin besar berarti semakin banyak terjadinya proses *crossover*.

VI. DAFTAR PUSTAKA

[1] D. E. Goldberg, Genetic Algorithm in Search, Optimization, and Machine Language, New York: Pearson Education, 2006.

[2] K. Henderson, Database Developer's Guide with Delphi 2, New York: Borland Press, 1996.

[3] B. Satyabudhi, "The Contribution of Computer Science Theories to Deterministic Hard-Operational Research Theories," Kristal, pp. 47-49, 1998.