PENJADWALAN PERKULIAHAN MENGGUNAKAN ALGORITME GENETIKA

Muhammad Syadid[1], Irman Hermadi[2], Sony Hartono Wijaya[2]

[1]Mahasiswa Departemen Ilmu Komputer, Fakultas Matematika dan IPA, Institut Pertanian Bogor

[2]Dosen Departemen Ilmu Komputer, Fakultas Matematika dan IPA, Institut Pertanian Bogor

Course scheduling is process of placing courses into available time-and-classroom slots. As number of course activities and requirement needed increase, the solution to scheduling problem become more complicated and time consuming. The aim of this research is to improve the previous research on Course Scheduling using Genetic Algorithm, such that number of students in each class complies with the class size. This research also used to measure the performance of Genetic Algorithm.

In this research, chromosomes are represented by a matrix of course activities. This matrix consists of all the course activities in one semester. Rows of matrix represent the amount of time-slots in a week course, from Monday to Saturday. Columns of the matrix represent the number of available classrooms.

The data used in this research are FMIPA IPB timetable data from Handbook of Undergraduate Program at IPB edition. The processing of this data produced students, lecturers, programs, time-slots, and course activities data. There are two kinds of students data used in this research, that is safe and unsafe data. The first data type ignores classroom and number of students of a class, while the latter does not.

The results of this research show that Genetic Algorithm gets the optimal solution with crossover rate 0.7, mutation rate 0.2, number of population 50, maximum generation 500, threshold 0, and stall generation 100. Results reveal that the number of conflicts on unsafe data is more than the safe one. The best schedule ever reached is by allocating of 6 time-slots per day with 6 classrooms with 97.02% effectivities and 93.33% efficiencies.

Keywords : course scheduling, genetic algorithms

PENDAHULUAN

Latar Belakang

Penjadwalan merupakan salah satu permasalahan pengalokasian aktivitas perkuliahan ke dalam slot waktu yang telah ditentukan. Lebih spesifik lagi penjadwalan perkuliahan merupakan masalah penempatan jadwal suatu aktivitas kuliah tertentu pada slot waktu dan ruang yang telah ditentukan. Penyelesaian masalah penjadwalan perkuliahan dalam jumlah yang sangat besar hingga saat ini masih menjadi permasalahan yang rumit untuk diselesaikan secara manual. Persyaratan-persyaratan yang harus dipenuhi menambah semakin kompleks dan rumitnya penyelesaian masalah penjadwalan. Semakin banyaknya persyaratan yang diajukan maka akan mempengaruhi lama waktu penyelesaian dan tingkat optimalitas *output* yang dihasilkan.

Algoritme Genetika merupakan algoritme pencarian yang didasarkan atas seleksi alami dan genetik yang pertama kali dikembangkan oleh John Holland dan mahasiswanya di University of Michigan (Goldberg 1989). Algoritme Genetika merupakan salah satu algoritme yang dapat menemukan solusi global yang paling optimal dalam ruang pencarian yang sangat besar dan kompleks. Algoritme Genetika dapat digunakan untuk menyelesaikan masalah penjadwalan perkuliahan.

Penelitian yang dilakukan oleh Tamba (2004) menghasilkan sebuah sistem yang mampu menyelesaikan masalah penjadwalan perkuliahan menggunakan Algoritme Genetika (studi kasus FMIPA IPB). Namun, penelitian yang dilakukan belum mengakomodasi masalah penjadwalan ruangan. Oleh karena itu, penelitian yang dilakukan merupakan tindak lanjut dari penelitian sebelumnya. Penelitian yang dilakukan, diharapkan dapat menerapkan Algoritme Genetika dengan menggunakan representasi kromosom yang berbeda sehingga dapat menyempurnakan kekurangan penelitian sebelumnya. Selain itu, penelitian ini juga diharapkan dapat memperoleh informasi berupa analisis kinerja dari Algoritme Genetika yang diterapkan pada masalah penjadwalan perkuliahan.

Tujuan

Tujuan dari penelitian ini adalah menyempurnakan penelitian sebelumnya agar jadwal yang dihasilkan dapat mengakomodasi masalah penjadwalan ruangan. Sekaligus menerapkan dan menganalisis kinerja dari Algoritme Genetika untuk masalah penjadwalan perkuliahan.

Ruang Lingkup

Ruang lingkup dari penelitian ini adalah :

1 Penelitian dibatasi pada penjadwalan perkuliahan mahasiswa *phasing out.*

2 Data yang digunakan dalam penelitian ini adalah data yang dibentuk secara manual.

3 Penelitian belum mengakomodasi masalah penjadwalan mahasiswa yang mengulang atau belum mengambil mata kuliah pada waktu yang telah ditentukan.

4 Satuan slot waktu terkecil untuk aktivitas perkuliahan adalah tiga SKS (Satuan Kredit Semester).

Manfaat

Penelitian ini diharapkan dapat membantu proses pembuatan jadwal perkuliahan yang mengakomodasi masalah penjadwalan ruangan dan memberikan informasi berupa analisis kinerja penerapan Algoritme Genetika dalam penyelesaian masalah penjadwalan perkuliahan.

PENJELASAN METODE

Masalah Penjadwalan Perkuliahan

Komponen dari masalah penjadwalan perkuliahan adalah :

- Kelompok dari mahasiswa
- Kelompok dari pengajar
- Kelompok dari mata kuliah
- Kelompok dari ruangan
- Kelompok dari slot waktu yang telah ditentukan

Aktivitas perkuliahan merupakan gabungan dari kelompok mahasiswa, dosen yang mengajar, dan mata kuliah yang diajarkan. Permasalahan yang sebenarnya dihadapi dalam kasus ini adalah permasalahan penempatan beberapa aktivitas perkuliahan yang ada ke dalam dimensi slot waktu dan ruangan yang telah ditentukan sehingga diperoleh solusi yang paling optimal. Solusi paling optimal dapat diperoleh ketika pelanggaran terhadap persyaratan-persyaratan yang diberikan berjumlah paling sedikit atau tidak ada.

Persyaratan Penjadwalan

Penjadwalan sangat berkaitan erat dengan persyaratan atau batasan-batasan yang dimiliki suatu institusi untuk menyelenggarakan sebuah kegiatan perkuliahan. Persyaratan-persyaratan tersebut adalah :

1 Hanya ada satu kegiatan perkuliahan yang dapat berlangsung dalam satu ruangan dalam satu waktu tertentu.
2 Setiap ruangan memiliki keterbatasan kapasitas daya tampung mahasiswa.
3 Setiap pengajar hanya mengajar satu kegiatan perkuliahan dalam satu waktu tertentu.
4 Satu kelompok mahasiswa hanya mengikuti satu kegiatan perkuliahan dalam satu waktu tertentu.

Kesulitan Masalah Penjadwalan

Kesulitan yang dihadapi dalam masalah penjadwalan adalah (Tamba 2004) :

1 Persyaratan khusus yang ditambahkan akan menambah lama waktu komputasi secara polinomial dalam pencarian solusi.
2 Perancangan metode heuristik yang efektif merupakan salah satu pekerjaan yang tidak mudah untuk dilakukan. Penggunaan prinsip heuristik untuk memotong ruang pencarian solusi yang tidak perlu, tidak dapat menjamin dapat menemukan solusi yang optimal atau mendekati optimal.
3 Tingkat visibilitas dari penjadwalan yang dihasilkan sangat dipengaruhi oleh beberapa persyaratan yang harus dipenuhi. Banyaknya persyaratan yang diajukan akan membuat masalah penjadwalan terlihat lebih kompleks dan sulit untuk diselesaikan.
4 Masalah penjadwalan sering terbentur dengan persyaratan di dunia nyata yang tidak dapat direpresentasikan dengan tepat ke dalam sistem.

Efektivitas dan Efisiensi Penjadwalan

Tingkat efektivitas penjadwalan berkaitan erat dengan jumlah pelanggaran yang dihasilkan terhadap jumlah aktivitas perkuliahan yang dijadwalkan. Tingkat efektivitas penjadwalan dapat dirumuskan sebagai berikut :

$$1 - \frac{Jumlah\ Pelanggaran}{Jumlah\ Aktivitas\ Perkuliahan} \times 100\%$$

Tingkat efisiensi penjadwalan berkaitan erat dengan jumlah sumber daya yang tersedia dengan jumlah aktivitas perkuliahan yang dijadwalkan. Sumber daya yang dimaksud adalah ketersediaan slot waktu dan ruangan. Tingkat efisiensi suatu jadwal dapat dirumuskan sebagai berikut :

$$\frac{Jumlah\ Aktivitas\ Perkuliahan}{Jumlah\ Sumber\ Daya} \times 100\%$$

Jadwal yang baik adalah sebuah jadwal yang memiliki nilai efektivitas dan efisiensi terbaik.

Algoritme Genetika

Algoritme Genetika merupakan bentuk algoritme pencarian solusi yang paling optimal berdasarkan pada mekanisme seleksi alam dan sifat-sifat genetika alamiah (Goldberg 1989). Kemunculan Algoritme Genetika banyak diinspirasikan oleh teori-teori yang ada dalam ilmu biologi, sehingga banyak istilah dan konsep-konsep ilmu biologi yang digunakan dalam Algoritme Genetika. Istilah-istilah biologi yang digunakan dalam Algoritme Genetika dapat dilihat pada Tabel 1.

Tabel 1 Istilah Biologi yang digunakan dalam Algoritme Genetika

Istilah Biologi dalam Algoritme Genetika	Definisi dalam Algoritme Genetika
Kromosom	Solusi dari suatu permasalahan
Gen	Bagian dari suatu solusi
Lokus	Posisi suatu gen dalam kromosom
Allele	Nilai dari suatu gen
Fenotip	Bentuk solusi dalam nilai sebenarnya
Genotip	Bentuk solusi dalam nilai yang disandikan
Crossover	Pindah silang/pertukaran string bit kromosom antar *parent*
Mutasi	Perubahan nilai string bit kromosom

Tahapan Algoritme Genetika

Tahapan dari Algoritme Genetika adalah (Lawrence 1991):

1 Inisialisasi populasi
2 Evaluasi populasi
3 Memilih anggota populasi yang terbaik untuk membentuk populasi baru atau disebut dengan proses seleksi.
4 Membentuk kromosom baru dengan cara rekombinasi dan mutasi
5 Evaluasi kromosom yang baru dan memasukkan ke dalam populasi.
6 Jika memenuhi kriteria *termination*, proses berhenti dan mengembalikan kromosom terbaik, jika belum maka kembali ke tahap dua.

Algoritme Genetika mulai melakukan pencarian solusi dengan melakukan inisialisasi populasi terlebih dahulu. Selanjutnya dilakukan proses evaluasi dan seleksi terhadap populasi yang terbentuk. Proses seleksi dilakukan berdasarkan fungsi *fitness* yang telah ditentukan. Individu yang memenuhi nilai *fitness* yang telah ditentukan akan bertahan hidup dan menentukan generasi berikutnya. Tahap selanjutnya, melalui operator rekombinasi dan mutasi, gen akan mengalami pengubahan materi genetik untuk membentuk generasi baru.

Operator rekombinasi bertugas menukar susunan genetik dari individu yang dipilih secara acak pada bagian titik potong tertentu. Operator mutasi bertugas melakukan pembalikan gen secara acak untuk membentuk individu baru. Satu iterasi dari Algoritme Genetika dianggap sebagai satu generasi.

Populasi ini selanjutnya secara terus menerus akan membentuk sebuah generasi yang lebih baik dari generasi-generasi sebelumnya. Hal ini disebabkan oleh kromosom yang memiliki nilai *fitness* relatif baik, akan bertahan hidup dan menjadi penerus untuk generasi-generasi selanjutnya. Sebaliknya, jika kromosom tersebut relatif buruk maka akan mati.

Komponen Utama Algoritme Genetika

Menurut Michalewicz (1996), Algoritme Genetika harus memiliki lima komponen berikut :

1 Representasi genetik dari setiap solusi yang mungkin dari suatu permasalahan.
2 Cara pembentukan populasi awal atau inisialisasi populasi
3 Fungsi evaluasi yang berperan menilai *fitness* dari solusi yang mungkin.
4 Operator genetik yang mengubah komposisi kromosom.
5 Nilai parameter yang digunakan dalam Algoritme Genetika, meliputi ukuran populasi, nilai probabilitas yang diterapkan dalam operator genetik (seleksi, *crossover*, mutasi).

Representasi Kromosom

Representasi kromosom dari solusi suatu permasalahan harus dilakukan sebelum dilakukan inisialisasi populasi. Hal ini disebut juga dengan *encoding*. *Encoding schema* merupakan cara menerjemahkan masalah ke dalam *framework* Algoritme Genetika.

Inisialisasi Populasi

Inisialisasi populasi merupakan salah satu tahapan awal yang paling penting dalam Algoritme Genetika agar menghasilkan solusi yang optimal. Inisialisasi populasi merupakan tahap pembentukan populasi yang dibentuk dari sekumpulan individu secara acak.

Fungsi *Fitness* (*Fitness Function*)

Suatu individu atau kromosom dievaluasi berdasarkan suatu fungsi tertentu sebagai ukuran performansinya. Fungsi yang digunakan untuk mengukur nilai kecocokan atau derajat optimalitas suatu kromosom disebut dengan *fitness function* (Suyanto 2005).

Nilai yang dihasilkan dari fungsi tersebut menandakan seberapa optimal solusi yang diperoleh. Nilai yang dihasilkan oleh fungsi *fitness* merepresentasikan seberapa banyak jumlah persyaratan yang dilanggar, sehingga dalam kasus penjadwalan perkuliahan semakin kecil jumlah pelanggaran yang dihasilkan maka solusi yang dihasilkan akan semakin baik.

Elitisme

Elitisme merupakan salah satu teknik dari Algoritme Genetika dalam mempertahankan kromosom yang memiliki nilai *fitness* terbaik untuk tetap bertahan hidup di generasi berikutnya. Elitisme dalam teknisnya dilakukan dengan meng-*copy* satu kromosom jika jumlah populasi ganjil, meng-*copy* dua kromosom jika jumlah populasi genap, atau dengan ketentuan nilai probabilitas tertentu.

Seleksi (*Selection*)

Seleksi merupakan proses pemilihan kromosom terbaik (nilai *fitness* terbaik) dari sebuah populasi untuk dilakukan proses selanjutnya yaitu rekombinasi dan mutasi. Salah satu metode seleksi yang banyak digunakan yaitu Roulette Wheel. Roulette Wheel menyeleksi populasi baru dengan distribusi probabilitas yang didasarkan pada nilai *fitness*. Tahapan dari seleksi Roulette Wheel adalah (Michalewicz 1996):

1 Hitung nilai *fitness eval*(v_i) untuk tiap kromosom v_i (i = 1, ..., *population_size*).

2 Hitung total *fitness* untuk populasi:

$$F = \sum_{i=1}^{pop\_size} eval(v_i))$$

3 Hitung peluang seleksi p_i untuk tiap kromosom v_i (i = 1, ..., *population_size*).

$$p_i = \frac{eval(v_i)}{F}$$

4 Hitung peluang kumulatif q_i untuk tiap kromosom v_i (i = 1, ..., *population_size*).

$$q_i = \sum_{j=1}^{i} p_j$$

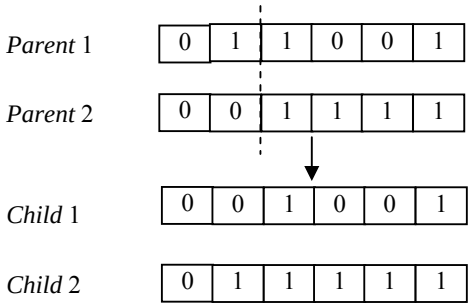
5 Proses seleksi dimulai dengan memutar Roulette Wheel sebanyak ukuran populasi (*population-size*). Pada sekali putaran, sebuah kromosom dipilih untuk membentuk populasi yang baru dengan cara berikut:

- Bangkitkan sebuah bilangan acak r pada selang [0, 1].
- Jika r < q_1 maka pilih kromosom pertama (v_1); selainnya pilih kromosom ke-i v_i ($2 \le i \le$ *population_size*) sehingga $q_{i-1} < r < q_i$.

Rekombinasi (*Crossover*)

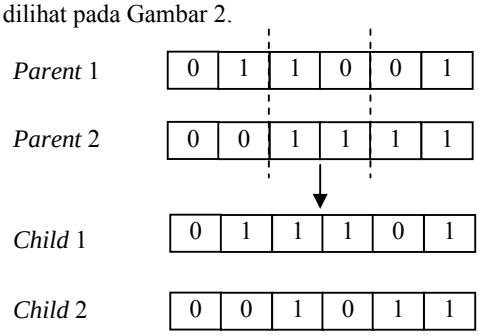
Crossover merupakan salah satu operator Algoritme Genetika yang sangat penting peranannya dalam menghasilkan sebuah solusi yang optimal. *Crossover* dapat dibedakan menjadi tiga jenis cara yang berbeda dalam menentukan titik segmen pertukaran string bit kromosomnya (Tamba 2004).

Pertama, *One-point Crossover* adalah sebuah cara penentuan titik segmen pertukaran string bit kromosom dengan memilih satu titik potong pertukaran. Titik potong dipilih secara acak, kemudian bagian pertama dari *parent* 1 digabungkan dengan bagian kedua *parent* 2. Skema *One-point crossover* dapat dilihat pada Gambar 1.



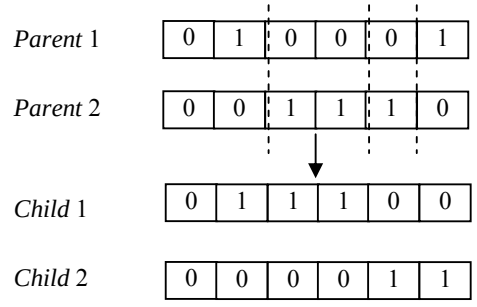
Gambar 1 *One-point crossover.*

Kedua, *Two-point crossover* yang merupakan perkembangan dari cara sebelumnya dengan memilih dua titik potong secara acak untuk melakukan pertukaran string bit kromosom. Skema *Two-point crossover* dapat dilihat pada Gambar 2.



Gambar 2 *Two-point crossover.*

Ketiga, jika jumlah gen dalam kromosom berjumlah sangat banyak maka penentuan titik potong pertukaran string bit kromosom dapat dilakukan dengan memilih lebih dari dua titik pertukaran yang dipilih secara acak. Cara yang ketiga ini disebut dengan *N-point Crossover*, n merupakan jumlah titik potong pertukaran yang dipilih secara acak. Skema *N-point crossover* dapat dilihat pada Gambar 3.



Gambar 3 *N-point crossover.*

Mutasi (*Mutation*)

Mutasi adalah salah satu operator Algoritme Genetika yang mengubah nilai gen pada kromosom untuk mendapatkan keturunan yang lebih baik dari generasi sebelumnya.

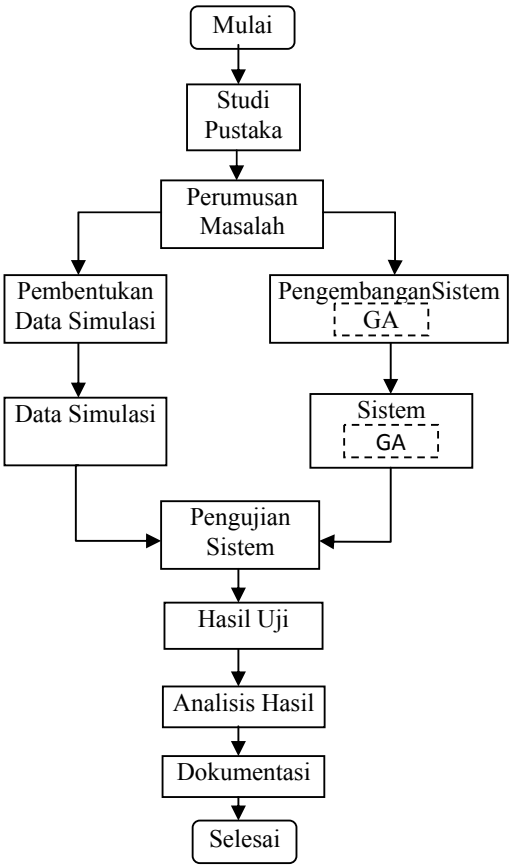
Termination

Tahap *termination* dari Algoritme Genetika dapat dilakukan dengan beberapa cara, antara lain :

- Menentukan jumlah generasi maksimum.
- Menentukan kriteria dari solusi yang diharapkan.
- Menentukan nilai *fitness* yang ingin dicapai.
- Kombinasi dari kondisi-kondisi di atas.

METODE PENELITIAN

Tahap-tahap dalam pengerjaan penelitian dilakukan sesuai dengan metode penelitian yang dapat dilihat pada Gambar 4.



Gambar 4 Skema metodologi penelitian.

Studi Pustaka

Pada tahap ini, dilakukan pengumpulan informasi dari buku, jurnal, dan artikel mengenai Algoritme Genetika dalam penyelesaian masalah penjadwalan perkuliahan. Literatur-literatur yang digunakan dapat dilihat pada bagian daftar pustaka.

Pembentukan Data Simulasi

Data yang digunakan dalam simulasi penelitian ini adalah data dari delapan departemen yang ada di FMIPA IPB. Delapan departemen tersebut adalah :

1 Departemen Statistika
2 Departemen Geofisika dan Meteorologi
3 Departemen Biologi
4 Departemen Kimia
5 Departemen Matematika
6 Departemen Ilmu Komputer
7 Departemen Fisika
8 Departemen Biokimia

Data yang digunakan berasal dari Buku Panduan Program Sarjana IPB edisi 2006 (revisi). Selanjutnya, data yang diperoleh diolah atau dilakukan pengkodean untuk didapatkan data mahasiswa, data pengajar, data mata kuliah, data slot waktu, dan data aktivitas perkuliahan.

Data kelompok mahasiswa dibentuk dari 8 departemen dari 3 angkatan (41, 42, dan 43) sehingga diperoleh 24 buah data mahasiswa. Penelitian ini menggunakan dua istilah dalam penentuan data mahasiswa, yaitu data mahasiswa *safe* dan data mahasiswa *unsafe*. Data mahasiswa *safe* adalah data mahasiswa yang mengabaikan kapasitas ruangan dan jumlah mahasiswa. Sedangkan data mahasiswa *unsafe* adalah data mahasiswa yang mempertimbangkan kapasitas ruangan dan jumlah mahasiswa. Data mahasiswa yang terbentuk dapat dilihat pada Lampiran 1.

Data pengajar dibentuk dengan pendekatan masing-masing departemen terdapat 7 pengajar sehingga jika terdapat 8 departemen maka diperoleh 56 buah data pengajar. Data pengajar yang terbentuk dapat dilihat pada Lampiran 2.

Data mata kuliah dibentuk dengan pendekatan bahwa mahasiswa dari masing-masing angkatan untuk setiap departemen mengambil 7 mata kuliah dalam satu semester, sehingga jika terdapat 24 kelompok mahasiswa maka akan terbentuk data mata kuliah sebanyak 168 buah. Data mata kuliah yang terbentuk dapat dilihat pada Lampiran 3.

Data slot waktu dalam penelitian ini dibuat sebanyak 3 macam. Pertama data slot waktu berjumlah 6 buah per hari sehingga dalam seminggu (Senin-Jumat) diperoleh data sebanyak 30 buah. Kedua, data berjumlah 7 buah per hari sehingga diperoleh 35 buah. Ketiga, data berjumlah 8 buah per hari sehingga diperoleh data sebanyak 40 buah. Data slot waktu dapat dilihat pada Lampiran 4, 5, dan 6. Data ruang dalam penelitian ini dibedakan menjadi 3 macam data, yaitu data ruang yang berjumlah 5, 6, dan 7 ruang. Data ruang yang terbentuk dapat dilihat pada Lampiran 7.

Pembentukan data aktivitas perkuliahan dalam penelitian ini dilakukan secara manual. Data dibuat agar

lebih mirip atau mendekati dengan keadaan sebenarnya. Pendekatan yang dilakukan, yaitu jumlah mata kuliah yang diambil oleh setiap angkatan mahasiswa dalam satu semester berjumlah 7 mata kuliah dan setiap pengajar mengajar sebanyak 3 mata kuliah di setiap departemen. Dari pendekatan yang dilakukan, sehingga diperoleh data aktivitas sebanyak 168 buah yang merupakan hasil penggabungan dari 24 data kelompok mahasiswa dengan 56 data pengajar dan 168 data mata kuliah. Data aktivitas perkuliahan yang terbentuk dapat dilihat pada Lampiran 8. Proses pengkodean dalam penelitian ini dilakukan secara manual oleh peneliti.

Pengembangan Sistem

Pengembangan sistem dimulai dengan pembuatan kode program sesuai dengan tahapan yang ada dalam Algoritme Genetika. Tahapan yang dilakukan adalah:

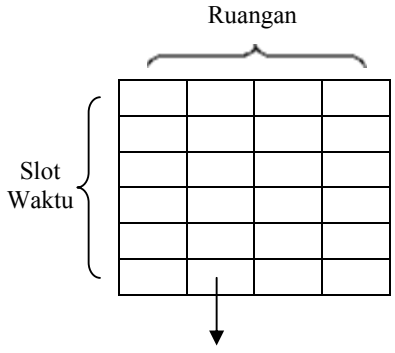
1 Representasi Kromosom

Kromosom pada penelitian ini direpresentasikan ke dalam bentuk matrik aktivitas perkuliahan. Matrik aktivitas perkuliahan adalah sebuah matrik yang berisi aktivitas perkuliahan yang terdiri dari mahasiswa, pengajar, dan mata kuliah yang diajarkan selama satu semester. Baris pada matrik aktivitas perkuliahan merepresentasikan jumlah slot waktu yang digunakan selama satu minggu perkuliahan (Senin – Jumat). Kolom pada matrik aktivitas perkuliahan merepresentasikan jumlah ruangan yang tersedia. Representasi kromosom dalam penelitian ini dapat dijelaskan pada Gambar 5.

Aktivitas perkuliahan dalam matrik aktivitas perkuliahan dikodekan sebagai berikut :

Mahasiswa : KOM41 (mahasiswa Ilmu Komputer angkatan 41)
Dosen : WAK (Wisnu Ananta Kusuma)
Mata kuliah : KOM271 (Basis Data)

Aktivitas perkuliahan dikodekan menjadi '&KOM41&&WAK&&KOM271&'.



Gambar 5 Representasi kromosom.

2 Inisialisasi Populasi

Populasi dibentuk dari *n* matrik aktivitas perkuliahan yang masing-masing matrik berisi seluruh aktivitas perkuliahan yang ditempatkan secara acak (*n* adalah ukuran populasi). Jika jumlah slot waktu ada 30 slot dan ruang ada 6 ruang maka matrik yang terbentuk akan berukuran 30x6 sebanyak *n* buah matrik.

3* Fungsi *fitness

Fitness akan dihitung berdasarkan jumlah pelanggaran yang terjadi pada setiap kromosom/matrik aktivitas kuliah. Pelanggaran yang dimaksud adalah :

1 Ketidaksesuaian antara jumlah mahasiswa dengan kapasitas ruangan.
2 Terdapat mahasiswa yang sama pada ruangan yang berbeda dalam satu slot waktu.
3 Terdapat pengajar yang sama pada ruangan yang berbeda dalam satu slot waktu.

Nilai *fitness* dalam penelitian ini dikatakan terbaik jika jumlah pelanggaran dari setiap kromosom diperoleh nilai yang paling kecil.

4 Elitisme

Jika jumlah populasi genap maka elitisme dalam penelitian ini dilakukan dengan meng-*copy* dua buah kromosom yang memiliki nilai *fitness* terbaik. Jika jumlah populasi ganjil maka hanya meng-*copy* satu buah kromosom yang memiliki nilai *fitness* terbaik. Kromosom elit akan bertahan hidup dalam generasi selanjutnya tanpa melalui proses rekombinasi (*Crossover*) dan mutasi (*Mutation*).

5 Seleksi (*Selection*)

Skema seleksi yang digunakan dalam penelitian ini adalah Roulette Wheel. Penggunaan skema ini memungkinkan kromosom yang memiliki nilai *fitness* terbaik akan memiliki peluang terpilih yang besar, sehingga populasi baru yang dihasilkan dimungkinkan akan terdiri dari sekumpulan kromosom yang memiliki nilai *fitness* terbaik.

6 Rekombinasi (*Crossover*)

Penelitian ini menerapkan *one-point crossover* sebagai metode rekombinasi. Proses penentuan titik potong dilakukan secara acak. Titik potong yang dimaksud merupakan indeks dari kolom matrik kromosom. Titik potong tersebut digunakan untuk memotong matrik berdasarkan kolom.

7 Mutasi (*Mutation*)

Mutasi dalam penelitian ini dilakukan dengan menukar posisi elemen matrik secara acak. Pertukaran dilakukan dengan menentukan dua buah posisi atau sepasang posisi elemen matrik secara acak untuk dilakukan pertukaran. Jumlah kromosom yang dimutasi ditentukan secara acak dengan nilai probabilitas tertentu.

8 *Termination*

Penelitian ini menggunakan maksimum generasi, nilai *threshold fitness,* dan *stall generation* sebagai kriteria berhentinya proses Algoritme Genetika. Algoritme Genetika akan berhenti ketika diperoleh sebuah individu yang memiliki nilai *fitness* sama dengan nilai *threshold* yang didefinisikan sebelumnya atau Algoritme Genetika akan berhenti ketika mencapai batas maksimum generasi. Algoritme Genetika juga akan berhenti ketika nilai *fitness* yang dihasilkan tidak berubah selama generasi tertentu (*stall generation*).

Pengujian dan Analisis

Pengujian dan analisis dalam penelitian ini dibagi menjadi beberapa tahap. Tahap pertama, yaitu untuk mengetahui kombinasi nilai parameter (*crossover,* mutasi, maksimum generasi, jumlah populasi, *threshold,* dan *stall generation*) terbaik sehingga menghasilkan jumlah pelanggaran yang paling kecil. Tahap kedua, yaitu untuk mengetahui perbedaan nilai *fitness* ketika digunakan data mahasiswa *safe* dan data mahasiswa *unsafe.* Tahap ketiga dilakukan untuk mengetahui pengaruh perubahan sumber daya yaitu ketersediaan slot waktu dan ruangan terhadap jumlah pelanggaran yang dihasilkan ketika digunakan data mahasiswa *unsafe.*

Lingkungan Pengembangan Sistem

Penelitian ini dikerjakan menggunakan perangkat keras komputer desktop dengan spesifikasi sebagai berikut :

- *Processor* : Intel Pentium Dual Core CPU 1.66 GHz,
- Memori : 960 MB, dan
- *Harddisk* : 80 GB.

Perangkat lunak yang digunakan dalam pengembangan sistem adalah :

- Sistem operasi : Windows XP,
- *Tools* : Matlab 7.0.1, dan
- Media penyimpanan data : Microsoft Excel 2003.

HASIL DAN PEMBAHASAN

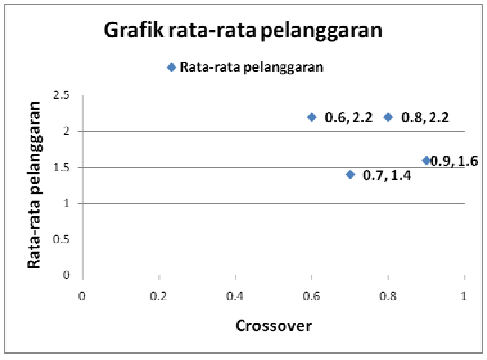
Pada penelitian ini, percobaan dilakukan sebanyak tiga kali. Percobaan pertama dilakukan untuk menentukan nilai parameter Algoritme Genetika yang paling optimal. Percobaan kedua dilakukan untuk mengetahui perbedaan hasil pelanggaran dari penggunaan data mahasiswa *safe* dan *unsafe.* Percobaan ketiga dilakukan untuk mengetahui pengaruh perubahan slot waktu dan ruang terhadap pelanggaran yang dihasilkan.

Percobaan dilakukan menggunakan aplikasi sederhana yang dibuat menggunakan Matlab 7.0.1. Antarmuka grafis dari aplikasi tersebut dapat dilihat pada Lampiran 14. *Output* dari aplikasi tersebut berupa kromosom jadwal yang tersimpan dalam file Microsoft Excel 2003. Kromosom jadwal dapat dilihat pada Lampiran 15. Selanjutnya, kromosom jadwal diolah menjadi jadwal yang lebih representatif agar mudah dipahami. Hasil olah kromosom jadwal dapat dilihat pada Lampiran 16.

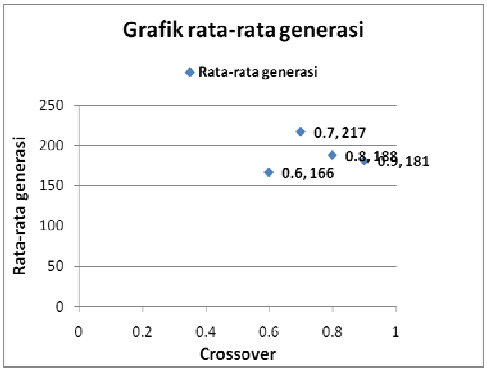
Percobaan 1

Percobaan pertama dilakukan untuk mengetahui kombinasi nilai parameter terbaik sehingga diperoleh jumlah pelanggaran yang paling sedikit. Percobaan dilakukan pengulangan masing-masing sebanyak 5 kali. Percobaan untuk mengetahui nilai parameter *crossover* terbaik dilakukan dengan mengubah nilai parameter *crossover*, sedangkan nilai parameter yang lain dibuat konstan. Begitu juga untuk memperoleh nilai parameter mutasi dan jumlah populasi, nilai parameter bukan uji dibuat konstan. Data yang digunakan adalah data aktivitas sebanyak 168 buah, data mahasiswa *safe,* data slot waktu 30 buah, dan data ruang 6 buah. Percobaan ini menggunakan nilai parameter mutasi, maksimum generasi, jumlah populasi, *threshold*, dan *stall generation* secara berurutan 0.1, 500, 50, 0, dan 100.

Nilai parameter *crossover* yang diuji adalah 0.6, 0.7, 0.8, dan 0.9. Hasil pengujian dapat dilihat pada Lampiran 9. Gambar 6 menunjukkan *plotting* nilai *crossover* dengan rata-rata pelanggaran yang dihasilkan, sedangkan Gambar 7 menunjukkan rata-rata generasi yang dicapai dalam memperoleh nilai *fitness* yang optimal. Hasil pengujian menunjukkan bahwa nilai parameter *crossover* yang paling optimal adalah 0.7.

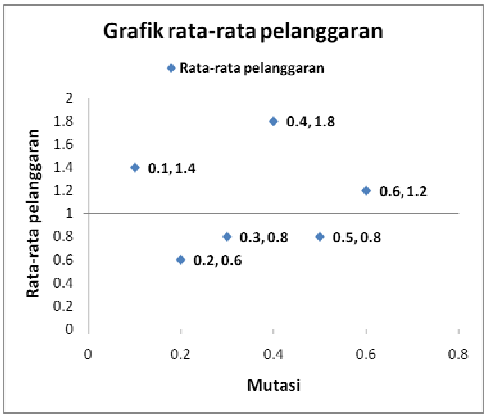


Gambar 6 Grafik rata-rata pelanggaran *crossover.*

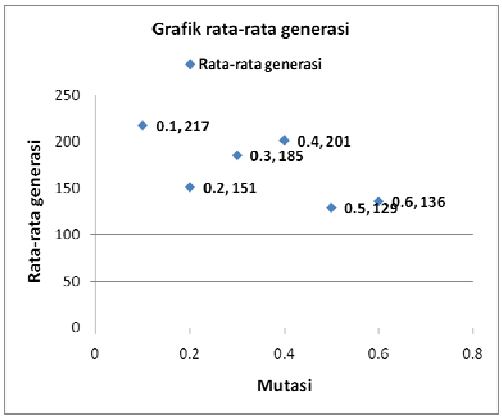


Gambar 7 Grafik rata-rata generasi *crossover.*

Nilai parameter mutasi yang diuji adalah 0.1, 0.2, 0.3, 0.4, 0.5, dan 0.6. Setiap pengujian dilakukan pengulangan sebanyak 5 kali dengan menggunakan nilai parameter *crossover* terbaik yang diperoleh pada percobaan sebelumnya, yaitu 0.7. Hasil pengujian dapat dilihat pada Lampiran 10. Hasil rata-rata pelanggaran dalam setiap pengujian dapat dilihat pada Gambar 8, sedangkan rata-rata generasi yang dicapai dalam setiap pengujian dapat dilihat pada Gambar 9. Hasil pengujian menunjukkan bahwa nilai mutasi yang paling optimal adalah 0.2.

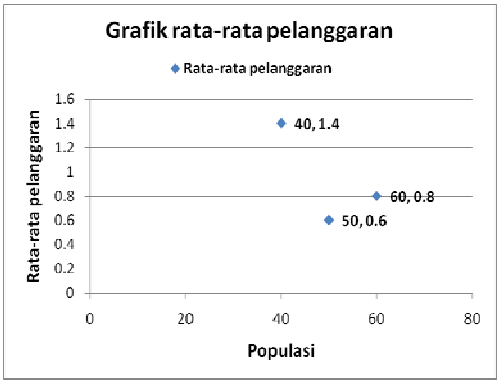


Gambar 8 Grafik rata-rata pelanggaran mutasi.

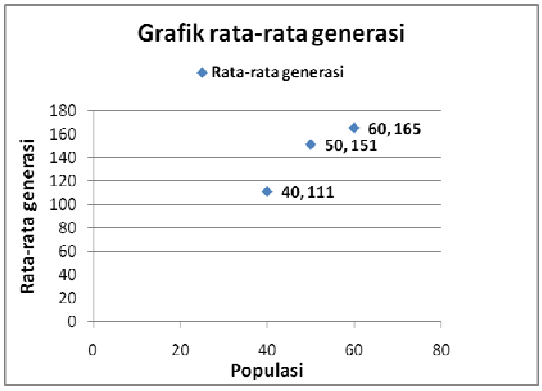


Gambar 9 Grafik rata-rata generasi mutasi.

Jumlah populasi yang diuji adalah 40, 50, dan 60. Setiap pengujian dilakukan pengulangan sebanyak 5 kali. Nilai parameter *crossover* dan mutasi yang digunakan adalah 0.7 dan 0.2. Hasil pengujian dapat dilihat pada Lampiran 11, sedangkan hasil rata-rata pelanggaran dan rata-rata generasi dari setiap pengujian dapat dilihat pada Gambar 10 dan Gambar 11. Hasil pengujian menunjukkan bahwa jumlah populasi yang paling optimal adalah 50.



Gambar 10 Grafik rata-rata pelanggaran populasi.



Gambar 11 Grafik rata-rata generasi populasi.

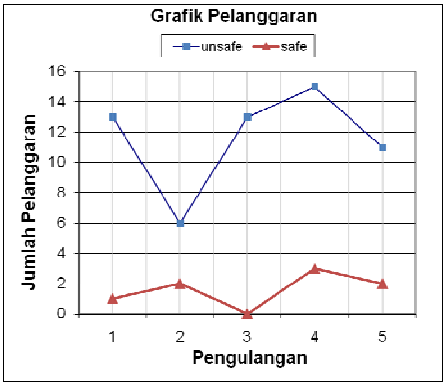
Penentuan maksimum generasi dilakukan tanpa pengujian karena dari percobaan sebelumnya dapat dilihat bahwa rata-rata nilai optimal dicapai di bawah 250 generasi, sehingga pemberian nilai 500 pada maksimum generasi tidak terlalu bermasalah. *Stall generation* yang digunakan adalah 100, penentuan nilai ini diambil berdasarkan nilai rata-rata selisih generasi pada satu pengulangan disaat terjadi perbaikan nilai *fitness*. Penentuan nilai *stall generation* yang terlalu kecil mengakibatkan algoritme berhenti terlalu cepat sehingga hasil yang diperoleh tidak optimal, sebaliknya nilai *stall generation* yang terlalu besar mengakibatkan proses semakin lama.

Percobaan 2

Percobaan kedua dilakukan dengan mengulang proses uji sebanyak 5 kali menggunakan data aktivitas, data slot waktu, dan data ruang yang sama dengan percobaan pertama. Nilai parameter yang digunakan pada *crossover*, mutasi, maksimum generasi, jumlah populasi, *threshold*, dan *stall generation* diambil dari percobaan pertama yang memiliki nilai *fitness* terbaik. *Crossover*, mutasi, maksimum generasi, jumlah populasi, *threshold*, dan *stall generation* yang digunakan secara berurutan bernilai 0.7, 0.2, 500, 50, 0, dan 100.

Percobaan kedua dilakukan untuk melihat perbedaan hasil yang diperoleh ketika digunakan data mahasiswa *safe* dan *unsafe*. Hasil percobaan menggunakan data mahasiswa *safe* menunjukkan bahwa dari 5 kali pengulangan jumlah rata-rata pelanggaran yang dihasilkan adalah 2 buah, sedangkan percobaan yang menggunakan data mahasiswa *unsafe* jumlah rata-rata pelanggaran yang dihasilkan sebanyak 12 buah. Hasil percobaan dapat dilihat pada Lampiran 12.

Hasil percobaan menunjukkan bahwa jumlah pelanggaran yang dihasilkan dari data mahasiswa *unsafe* lebih besar daripada data mahasiswa *safe*. Grafik pelanggaran yang dihasilkan dari data mahasiswa *safe* dan data mahasiswa *unsafe* dapat dilihat pada Gambar 12.



Gambar 12 Grafik pelanggaran data mahasiswa *safe* dan *unsafe*.

Percobaan 3

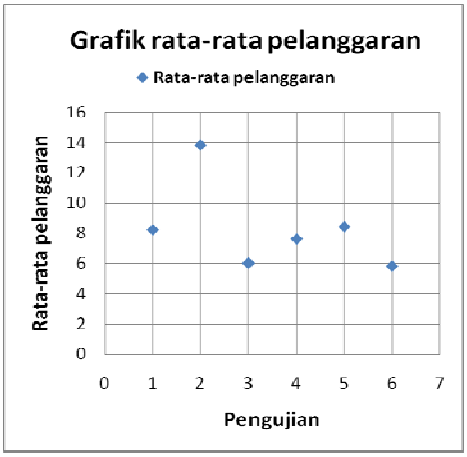
Percobaan yang dilakukan sebelumnya menunjukkan bahwa jumlah pelanggaran yang dihasilkan dari data mahasiswa *unsafe* lebih besar daripada data mahasiswa *safe*. Hal ini kemungkinan dipengaruhi oleh ketersedian sumber daya yang ada atau lebih tepatnya ketersediaan slot waktu dan ruang. Percobaan ketiga dilakukan untuk mengetahui pengaruh perubahan sumber daya yang ada yaitu ketersediaan slot waktu dan ruangan terhadap jumlah pelanggaran yang dihasilkan ketika digunakan data mahasiswa *unsafe*. Percobaan ini akan menghasilkan kombinasi jumlah slot waktu dan ruangan yang tepat sehingga menghasilkan sebuah jadwal yang memiliki tingkat efisiensi dan efektivitas yang tinggi.

Percobaan dilakukan dengan melakukan 6 kali uji dan setiap uji dilakukan 5 kali pengulangan. Pada percobaan kali ini seluruh nilai parameter dan data dibuat konstan kecuali data slot waktu dan data ruangan. Nilai parameter yang digunakan adalah 0.7 untuk *crossover,* 0.2 untuk mutasi, 500 untuk maksimum generasi, 50 untuk jumlah populasi, 0 untuk *threshold*, dan 100 untuk nilai *stall generation.* Data aktivitas yang digunakan berjumlah 168 buah dan data mahasiswa berjumlah 24 buah (masing-masing 3 angkatan dari 8 departemen), sedangkan data slot waktu dan data ruang yang digunakan dalam setiap uji dapat dilihat pada Tabel 2.

Tabel 2 Tabel uji

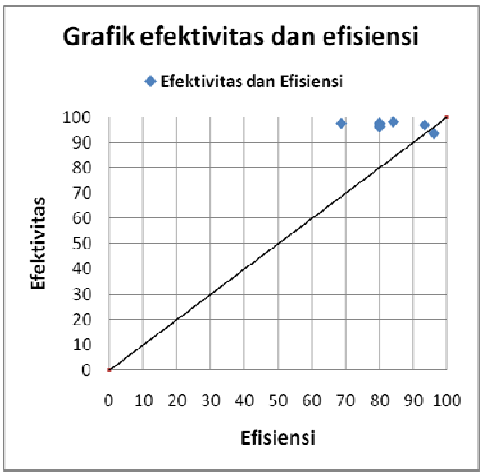
Uji	I	II	III	IV	V	VI
Slot waktu per hari	6	7	7	8	6	7
Slot waktu per minggu	30	35	35	40	30	35
Jumlah ruang	6	5	6	5	7	7
Jumlah sumber daya	180	175	210	200	210	145

Hasil uji pertama menunjukkan bahwa nilai *fitness* terbaik yang dicapai pada ulangan ke-4 berjumlah 5 buah pada generasi ke-500 dengan lama waktu proses 7974.84 detik. Hasil dari 5 kali pengulangan rata-rata nilai *fitness* yang diperoleh sebesar 8. Hasil uji kedua nilai *fitness* terbaik dicapai pada ulangan ke-5 pada generasi ke-391 dengan jumlah pelanggaran sebanyak 11 buah dan lama proses 5560.81 detik, sedangkan rata-rata nilai *fitness* yang diperoleh sebesar 14. Hasil uji ketiga nilai *fitness* terbaik dicapai pada ulangan ke-3 dengan jumlah pelanggaran sebanyak 4 buah dicapai pada generasi ke-347 dengan lama proses 5939.56 detik, sedangkan rata-rata pelanggaran yang diperoleh sebesar 6. Hasil uji keempat nilai *fitness* terbaik dicapai pada ulangan ke-1 dengan jumlah pelanggaran sebanyak 3 buah pada generasi ke-500 dengan lama proses 7277.56 detik, sedangkan rata-rata nilai *fitness* yang diperoleh sebesar 8. Hasil uji kelima nilai *fitness* terbaik dicapai pada ulangan ke-3 dengan jumlah pelanggaran sebanyak 6 buah pada generasi ke-269 dengan lama proses 5052.73 detik, sedangkan rata-rata nilai *fitness* yang diperoleh sebesar 8. Hasil uji keenam nilai *fitness* terbaik dicapai pada ulangan ke-1 dan ke-2 dengan jumlah pelanggaran sebanyak 4 buah. Pada ulangan ke-1 dicapai pada generasi ke-300 dengan lama proses 5938.63 detik, sedangkan pada ulangan ke-2 dicapai pada generasi ke-277 dengan lama proses 3722.47 detik. Rata-rata pelanggaran yang dihasilkan sebesar 6. Hasil uji keseluruhan dapat dilihat pada Lampiran 13. Nilai rata-rata pelanggaran dari seluruh pengujian adalah sebesar 8.3 yang ditunjukkan berupa garis lurus pada grafik. Hubungan antara rata-rata nilai *fitness* yang dihasilkan terhadap masing-masing pengujian dapat dilihat pada Gambar 13.



Gambar 13 Grafik rata-rata pelanggaran.

Hasil uji pertama menunjukkan bahwa tingkat efektivitas yang dicapai sebesar 97.02%, sedangkan tingkat efisiensi yang dicapai sebesar 93.33%. Hasil uji kedua memiliki tingkat efektivitas sebesar 93.45% dan tingkat efisiensi sebesar 96%. Hasil uji ketiga menunjukkan bahwa tingkat efektivitas yang dicapai sebesar 97.62%, sedangkan tingkat efisiensi yang dicapai sebesar 80%. Hasil uji keempat memiliki tingkat efektivitas sebesar 98.21% dan tingkat efisiensi sebesar 84%. Hasil uji kelima menunjukkan bahwa tingkat efektivitas yang dicapai sebesar 96.43%, sedangkan tingkat efisiensi yang dicapai sebesar 80%. Hasil uji keenam memiliki tingkat efektivitas sebesar 97.62% dan tingkat efisiensi sebesar 68.57%. Hubungan antara tingkat efektivitas dan efisiensi dari masing-masing pengujian dapat dilihat pada Gambar 14.



Gambar 14 Grafik efektivitas dan efisiensi.

Hasil jadwal terbaik adalah jadwal yang memiliki nilai efektivitas dan efisiensi yang cukup tinggi dan memiliki perbedaan nilai efektivitas dan efisiensi yang tidak terlalu besar. Pada pengujian pertama diperoleh nilai efektivitas dan efisiensi yang cukup tinggi dan selisih yang tidak terlalu besar, sehingga jadwal yang dihasilkan pada pengujian pertama merupakan hasil yang terbaik. Pada pengujian pertama nilai efektivitas yang dihasilkan sebesar 97.02% dan nilai efisiensi yang dihasilkan sebesar 93.33%.

KESIMPULAN DAN SARAN

Kesimpulan

Penelitian yang dilakukan telah berhasil menyempurnakan kekurangan pada penelitian sebelumnya dalam hal pengakomodasian masalah penjadwalan ruangan. Penelitian juga telah berhasil menerapkan Algoritme Genetika sebagai metode untuk menyelesaikan masalah penjadwalan perkuliahan.

Hasil dari penelitian ini menunjukkan bahwa kinerja Algoritme Genetika mencapai nilai optimal pada kombinasi nilai parameter *crossover* 0.7, mutasi 0.2, jumlah populasi 50, maksimum generasi 500, *threshold* 0, dan *stall generation* 100. Hasil percobaan menunjukkan bahwa jumlah pelanggaran yang dihasilkan dari data mahasiswa *unsafe* lebih besar daripada data mahasiswa *safe*. Hasil percobaan pengalokasian 6 slot waktu per hari dan 6 ruang menghasilkan jadwal terbaik dengan nilai efektivitas sebesar 97.02% dan efisiensi sebesar 93.33%.

Saran

Penelitian ini dapat dikembangkan lagi untuk mengakomodasi masalah penjadwalan perkuliahan Mayor-Minor dan praktikum. Proses eksekusi Algoritme Genetika dapat dipercepat dengan menerapkan Algoritme Genetika Paralel.

DAFTAR PUSTAKA

Burke E, Elliman D, Weare R. 1994. *A Genetic Algorithm Based University Timetabling System.* Department of Computer Science. University of Nottingham.

Goldberg D. 1989. *Genetic Algorithm in Search, Optimization, and Machine Learning.* New York : Addison-Wesley Publishing Company.

Lawrence D. 1991. *Handbook of Genetic Algorithms.* New York : Van Nostrand Reinhold.

Michalewicz Z. 1996. *Genetic Algorithms + Data Structure = Evolution Programs*. New York: Springer-Verlag Berlin Heidelberg.

Suyanto**.** 2005. *Algoritma Genetika dalam Matlab.* Yogyakarta : Penerbit Andi.

Tamba GMP. 2004. *Sistem Penjadwalan Perkuliahan Menggunakan Algoritma Genetika (Studi Kasus Fakultas Matematika dan IPA IPB)* [skripsi]. Bogor : Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Institut Pertanian Bogor.