PERBANDINGAN PERFORMANSI ALGORITMA GENETIKA DAN ALGORITMA *ANT COLONY OPTIMIZATION* DALAM OPTIMASI PENJADWALAN MATA KULIAH

Skripsi

disusun sebagai salah satu syarat

untuk memperoleh gelar Sarjana Sains

Program Studi Teknik Informatika

oleh

Imam Ahmad Ashari
4611412015

JURUSAN ILMU KOMPUTER

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS NEGERI SEMARANG

2016

PERNYATAAN

Saya menyatakan bahwa skripsi ini bebas plagiat, dan apabila di kemudian hari terbukti terdapat plagiat dalam skripsi ini, maka saya bersedia menerima sanksi sesuai ketentuan peraturan perundang-undangan.

Semarang, 24 Agustus 2016

Imam Ahmad Ashari

4611412015

PERSETUJUAN PEMBIMBING

Nama : Imam Ahmad Ashari

Nim : 4611412015

Program Studi : S-1 Teknik Informatika

Judul Skripsi : Perbandingan Performansi Algoritma Genetika dan Algoritma *Ant Colony Optimization* dalam Optimasi Penjadwalan Mata Kuliah

Skripsi ini telah disetujui oleh pembimbing untuk diajukan ke sidang panitia ujian skripsi Program Studi Teknik Informatika FMIPA UNNES.

Semarang, 24 Agustus 2016

Pembimbing 1	Pembimbing 2
Much Aziz Muslim, S.Kom., M.Kom.	Alamsyah, S.Si., M.Kom.
NIP. 197404202008121001	NIP. 197405172006041001

PENGESAHAN

Skripsi yang berjudul

Perbandingan Performansi Algoritma Genetika dan Algoritma *Ant Colony Optimization* dalam Optimasi Penjadwalan Mata Kuliah

disusun oleh

Imam Ahmad Ashari

4611412015

telah dipertahankan di hadapan sidang Panitia Ujian Skripsi FMIPA UNNES pada tanggal 24 Agustus 2016.

Panitia:

Ketua

Prof. Dr. Zaenuri, S.E., M.Si., Akt.
NIP. 196412231988031001

Sekretaris

Endang Sugiharti, S.Si., M.Kom.
NIP. 197401071999032001

Ketua Penguji

Riza Arifudin, S.Pd., M.Cs.
NIP. 198005252005011001

Anggota Penguji/
Pembimbing I

Much Aziz Muslim, S.Kom., M.kom.
NIP. 197404202008121001

Anggota Penguji/
Pembimbing II

Alamsyah, S.Si., M.Kom.
NIP. 197405172006041001

MOTTO DAN PERSEMBAHAN

MOTTO

“Keajaiban itu nyata! Lahir saat kita berusaha dan bekerja keras”. (Penulis)

“Membahagiakan diri sendiri dan membahagiakan semua orang terdekat adalah prinsip hidup yang utama”.(Penulis)

“Kita tidak hidup dilingkaran ruang yang kecil melainkan kita hidup dilingkaran ruang yang sangat besar , jadi jangan berhenti disatu poros saja”. (Penulis)

PERSEMBAHAN

Skripsi ini ku persembahkan kepada:

1. Orang tua tercinta terimakasih atas doa, dukungan dan kasih sayang yang tiada hentinya engkau berikan.
2. Adik saya yang saya sayangi, Didik Olfaya dan Nizar Zulmi.
3. Saudara saya yang saya selalu memotivasi dan senantiasa menasehati serta memberikan saran dan masukan.
4. Sahabat terdekat yang telah memberikan kesan dalam penulisan skripsi ini.
5. Almamaterku UNNES.

PRAKATA

Puji syukur penulis panjatkan kehadirat Allah SWT yang telah melimpahkan segala rakhmat dan hidayah-Nya dalam penyusunan skripsi, sehingga penulis dapat menyelesaikan skripsi dengan judul "**Perbandingan Performansi Algoritma Genetika dan Algoritma *Ant Colony Optimization* dalam Optimasi Penjadwalan Mata Kuliah**".

Skripsi ini dapat diselesaikan karena adanya kerjasama, bantuan dan motivasi dari berbagai pihak. Ucapan terima kasih ini penulis tujukan kepada yang terhormat:

1. Prof. Dr. Fathur Rokhman, M.Hum., Rektor Universitas Negeri Semarang, yang telah memberikan kesempatan kepada penulis untuk dapat berkuliah di Jurusan Ilmu Komputer Program Studi Teknik Informatika FMIPA UNNES;
2. Prof. Dr. Zaenuri, S.E., M.Si., Akt., Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang yang telah memberikan izin kepada penulis untuk menyusun skripsi;
3. Endang Sugiharti, S.Si., M.Kom., Ketua Jurusan Ilmu Komputer yang telah banyak memberi bimbingan kepada penulis;
4. Riza Arifudin, S.Pd, M.Cs., selaku ketua penguji, yang telah memberikan ijin kepada penulis untuk menyusun skripsi, serta memberikan banyak masukan, kritik dan saran dalam penyelesaian skripsi ini;

5. Much Aziz Muslim, S.Kom., M.Kom., Dosen Pembimbing I yang telah meluangkan banyak waktu, membantu, membimbing, dan mengarahkan untuk memberikan bimbingan pada penulis dalam menyelesaikan skripsi;
6. Alamsyah, S.Si., M.Kom., Dosen Pembimbing II yang telah meluangkan banyak waktu, membantu, membimbing, dan mengarahkan untuk memberikan bimbingan pada penulis dalam menyelesaikan skripsi;
7. Ayahanda dan Ibunda tercinta serta adikku tersayang, yang telah memberikan do'a dan dorongan baik secara moril, materil maupun spiritual dalam menyelesaikan skripsi;
8. Sahabat dan rekan-rekan Ilmu Komputer 2012 yang bersama-sama berjuang dalam menyelesaikan skripsi;
9. Seluruh staf dosen di Universitas Negeri Semarang.
10. Serta semua pihak yang tidak dapat penulis sebutkan satu persatu disini, terima kasih atas bantuan dan dorongannya.

Semoga bantuan yang telah diberikan kepada kepada penulis mendapatkan imbalan dari Allah Yang Maha Pengasih.

Semarang, 24 Agustus 2016

Imam Ahmad Ashari

ABSTRAK

Ashari, Imam Ahmad. 2016. *Perbandingan Performansi Algoritma Genetika dan Algoritma Ant Colony Optimization dalam Optimasi Penjadwalan Mata Kuliah*. Skripsi, Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang. Pembimbing Utama Much Aziz Muslim, S.Kom., M.Kom. dan Pembimbing Pendamping Alamsyah, S.Si., M.Kom.

Kata kunci: Penjadwalan Mata Kuliah, Algoritma Genetika, Algoritma *Ant Colony Optimization*, Algoritma *Metaheuristic*, Performansi.

Masalah penjadwalan di universitas merupakan jenis masalah penjadwalan yang rumit. Proses penjadwalan harus dilakukan di setiap pergantian semester, hal tersebut menjadikan pekerjaan ini terasa melelahkan dan memakan banyak waktu. Dalam masalah penjadwalan di universitas setiap batasan tidak boleh dilanggar

Algoritma *metaheuristic* merupakan algoritma yang cocok untuk menyelesaikan permasalahan penjadwalan mata kuliah. Algoritma *metaheuristic* merupakan algoritma yang memiliki banyak cara dalam menyelesaikan permasalahan sampai ke batas solusi optimal. Pada penelitian ini akan dilakukan perbandingan antara algoritma *ant colony optimization* dengan algoritma genetika yang merupakan algoritma evolusi untuk menyelesaikan sebuah permasalahan penjadwalan mata kuliah.

Tujuan dari penelitian ini adalah untuk mengetahui hasil perbandingan performansi yang lebih baik antara algoritma genetika dan algoritma *ant colony optimization* dalam menyelesaikan jadwal mata kuliah pada solusi terbaik

Hasil dari penelitian ini adalah algoritma genetika mempunyai performansi lebih baik dibandingkan algoritma *ant colony optimization* dalam menyelesaikan kasus penjadwalan mata kuliah. Algoritma genetika mendapatkan solusi terbaik lebih cepat dan lebih sedikit memakan memori dibandingkan dengan algoritma *ant colony optimization* saat melakukan proses komputasi.

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel

Halaman

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang

Jadwal adalah serangkaian pertemuan pada waktu yang bersamaan. Pertemuan disini adalah pertemuan yang memadukan berbagai sumber daya seperti orang, ruangan dan lain-lain (Jain dkk., 2010: 248). Masalah penjadwalan merupakan salah satu masalah optimasi kombinatorial yang umum terjadi di kehidupan sehari-hari, masalah penjadwalan dalam sektor pendidikan bukanlah hal yang baru. Secara umum, masalah penjadwalan dapat diklasifikasikan menjadi beberapa jenis, seperti penjadwalan tingkat akademis Perguruan Tinggi, penjadwalan Sekolah Dasar dan Menengah, penjadwalan ujian, penjadwalan transportasi, penjadwalan penjualan atau pengiriman barang dan lain-lain (Nugraha & Kosala, 2014: 789).

Masalah penjadwalan di universitas merupakan jenis masalah penjadwalan yang rumit. Proses penjadwalan harus dilakukan di setiap pergantian semester, hal tersebut menjadikan pekerjaan ini terasa melelahkan dan memakan banyak waktu. Hal yang perlu diperhatikan dalam membuat jadwal disebuah universitas adalah melakukan alokasi seluruh kegiatan di *timeslots* dan ruangan dengan memperhatikan daftar batasan yang diberikan oleh universitas dalam satu semester, sehingga tidak ada konflik yang terjadi dalam alokasi tersebut. Dalam

masalah penjadwalan di universitas setiap batasan tidak boleh dilanggar (Babaei, 2015: 2).

Alamsyah (2004) menjelaskan bahwa inti dari masalah penjadwalan mata kuliah di universitas adalah karena banyaknya komponen yang perlu diperhatikan dalam pembuatan jadwal, beberapa komponen itu terdiri dari mahasiswa, dosen, waktu dan ruang dengan memperhatikan batasan dan syarat tertentu sehingga tidak terjadi tumbukan dalam jadwal seperti tumbuk ruangan, tumbuk dosen pengajar dan lain-lain.

Untuk menyelesaikan sebuah permasalahan penjadwalan teknik yang paling tepat digunakan adalah teknik optimasi. Teknik optimasi dapat memberikan hasil terbaik yang diinginkan. Teknik optimasi memerlukan strategi yang bagus dalam mengambil keputusan agar diperoleh hasil yang optimum. Untuk itu, dibutuhkan metode optimasi yang dapat diterapkan untuk menyusun jadwal perkuliahan dengan beberapa batasan dan aturan yang ada.

Metode evolusi merupakan salah satu metode optimasi yang sering diteliti beberapa tahun terakhi. Squillero (2015: 1) menjelaskan bahwa metode evolusi adalah metode alam yang terinspirasi dari seleksi alam yang menyebabkan variasi untuk dikumpulkan dalam satu arah tertentu yang memperlihatkan hasil yang baik sehingga menyerupai proses optimasi yang disengaja. Memang proses tersebut hanya melihat efek dari perubahan secara acak, tetapi dari hal itu peneliti terinspirasi untuk membuat perhitungan evolusi di cabang ilmu komputer dari kecerdasan komputasi yang berfokus pada algoritma acak yang mendasari dari

teori evolusi. Perhitungan evolusi tersebut termasuk ke dalam kerangka kerja *metaheuristic*.

Beberapa tahun terakhir ini, ilmuwan telah banyak melakukan penelitian tentang Algoritma Genetika (GA), untuk memecahkan masalah optimasi yang melibatkan tujuan fungsi tunggal untuk memecahkan masalah *multiobjective* (Singh dkk., 2012: 48). Selain Algoritma Genetika juga terdapat Algoritma *metaheuristic* lainnya yang banyak digunakan penelitian dalam menyelesaikan optimasi penjadwalan, seperti algoritma *ant colony optimization*, algoritma *quantum genetic*, *simulated annealing* dan *particle swarm optimization* (Saragih dkk., 2012: 78).

Salah satu algoritma *metaheuristic* yang biasa digunakan untuk menyelesaikan sebuah masalah optimasi adalah *algoritma ant colony optimization* atau yang biasa disebut algoritma semut. Algoritma *ant colony optimization* merupakan algoritma yang awalnya terinspirasi dari perilaku semut, dari algoritma *ant colony optimization* banyak dikembangkan jenis algoritma lainnya seperti: algoritma *Ant System*, algoritma *Elitist ant System*, algoritma *Rank-Base Ant System*, algoritma *Max-Mint Ant System* (MMAS) dan algoritma *Ant Colony System* (Tiwari & Vidyarthi, 2016: 78).

Pada penelitian ini akan dilakukan perbandingan antara algoritma *ant colony optimization* dengan algoritma genetika yang merupakan algoritma evolusi untuk menyelesaikan sebuah permasalahan penjadwalan mata kuliah, dari kedua algoritma akan dibandingkan mana algoritma yang mempunyai performansi yang lebih baik dalam menyelesaikan jadwal mata kuliah pada solusi terbaik.

Objek yang akan digunakan untuk membandingkan kedua algoritma ini adalah jadwal mata kuliah di Jurusan Ilmu Komputer pada semester gasal tahun ajaran 2015/2016.

Berdasarkan latar belakang permasalahan tersebut, maka penelitian ini mengambil judul "**Perbandingan Performansi Algoritma Genetika dan Algoritma *Ant Colony Optimization* dalam Optimasi Penjadwalan Mata Kuliah**".

1.2 Rumusan Masalah

Dari uraian latar belakang masalah di atas dapat dirumuskan permasalahan yang ada yaitu bagaimana hasil perbandingan performansi komputasi yang lebih baik antara algoritma genetika dan algoritma *ant colony optimization* dalam menyelesaikan jadwal mata kuliah pada solusi terbaik?

1.3 Batasan Masalah

Adapun batasan dalam penelitian ini adalah sebagai berikut:

1) Algoritma yang dipakai dalam optimasi penjadwalan mata kuliah adalah algoritma genetika dan algoritma *ant colony optimization*.
2) Parameter perbandingan performansi yang digunakan dalam penelitian adalah waktu eksekusi dan memori yang digunakan dari kedua algoritma saat melakukan optimasi penjadwalan mata kuliah pada solusi terbaik.
3) Objek yang digunakan sebagai data penelitian adalah data jadwal mata kuliah di Jurusan Ilmu Komputer di Universitas Negeri Semarang pada semester gasal tahun ajaran 2015/2016.
4) Data jadwal mata kuliah yang digunakan hanya data dari mata kuliah prodi.

5) Optimasi penjadwalan mata kuliah akan dirancang dengan bahasa pemrograman PHP, *Software Sublime* 3.0 (Sebagai *Text Editor*), *Database Management System* (DBMS) *MySQL Xampp*.

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian dengan membandingkan performansi antara algoritma genetika dan algoritma *ant colony optimization* adalah untuk mengetahui algoritma mana yang mempunyai performansi komputasi lebih baik dalam menyelesaikan masalah penjadwalan mata kuliah.

1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah untuk memberikan pengetahuan tentang performansi komputasi algoritma mana yang lebih baik antara algoritma genetika dan algoritma *ant colony optimization* dalam menyelesaikan masalah penjadwalan mata kuliah.

1.6 Sistematika Skripsi

Sistematika penulisan untuk memudahkan dalam memahami alur pemikiran secara keseluruhan skripsi. Penulisan skripsi ini secara garis besar dibagi menjadi tiga bagian yaitu sebagai berikut:

1) Bagian Awal Skripsi

Bagian awal skripsi terdiri dari halaman judul, halaman pengesahan, halaman pernyataan, halaman motto dan persembahan, abstrak, kata pengantar, daftar isi, daftar gambar, daftar tabel dan daftar lampiran.

2) Bagian Isi Skripsi

Bagian isi skripsi terdiri dari lima bab yaitu sebagai berikut.

a. Bab 1: Pendahuluan

Bab ini terdiri atas latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian serta sistematika skripsi.

b. Bab 2: Tinjauan Pustaka

Bab ini terdiri dari atas landasan teori, contoh kasus dan penelitian terkait.

c. Bab 3: Metode Penelitian

Bab ini terdiri atas Studi Pendahuluan, Tahap Pengumpulan Data, Tahap Pengembangan Sistem Uji Metode dan Perancangan Aplikasi (*Design*).

d. Bab 4: Hasil dan Pembahasan

Bab ini terdiri atas hasil penelitian dan pembahasan penelitian.

e. Bab 5: Pentup

Bab ini terdiri atas simpulan dan saran

3) Bagian Akhir Skripsi

Bagian akhir skripsi berisi daftar pustaka yang merupakan informasi mengenai buku-buku, sumber-sumber dan referensi yang digunakan penulis serta lampiran-lampiran yang mendukung dalam penulisan skripsi ini.

BAB II

TINJAUAN PUSTAKA

2.1 Algoritma Genetika

2.1.1 Definisi Algoritma Genetika

Algoritma genetika ditemukan oleh John H. Holland dari *university of Michigan* yang memulai penelitiannya pada tahun 1960. Dalam penelitian Holland penerapan algoritma genetika dikaitkan dengan metode adaptif untuk memecahkan masalah dan optimasi.

Algoritma genetika didasarkan pada proses kelangsungan makhluk hidup pada setiap generasi dalam sebuah populasi secara alami. Proses seleksi tersebut mencerminkan sebuah proses seleksi alam yang secara bertahap akan berjalan mengikuti alam seperti siapa kuat dia yang akan bertahan. Dengan meniru proses tersebut, algoritma genetika dapat digunakan untuk menyelesaikan permasalahan-permasalah dalam dunia nyata dengan menghasilkan solusi yang paling baik (Beasly & Chu, 1994).

Algoritma genetika banyak digunakan untuk menyelesaikan berbagai masalah optimasi. Algoritma genetika bukanlah algoritma yang proses penyelesaiannya berdasarkan perhitungan matematika melainkan proses penyelesaiannya memanfaatkan analogi mekanisme seleksi alam dan mekanisme kawin silang, mutasi, inversi, dan lain-lain yang terdapat pada proses genetika alam.

Dalam algoritma genetika sebuah solusi dinyatakan sebagai kromosom. Dalam algoritma genetika tahap awal kromosom dibangkitkan secara acak dalam sebuah populasi. Dari setiap kromosom akan dievaluasi tingkat kefektifannya dengan menghitung nilai *fitness* dari masing masing kromosom yang telah dibangkitkan. Semakin tinggi nilai *fitness* kromosom tersebut maka peluang terpilihnya kromosom tersebut akan semakin tinggi.

2.1.2 Kelebihan dan Kekurangan Algoritma Genetika

Algoritma genetika mempunyai beberapa kelebihan dan kekurangan dibandingkan dengan algoritma lainnya, Berlianty & Arifin (2010: 119-120) menjelaskan tentang kelebihan dan kekurangan algoritma genetika, yaitu sebagai berikut:

Kelebihan algoritma genetika

1) Algoritma genetika bekerja dengan memanipulasi kode-kode set parameter, bukan dengan hasil manipulasi nilai parameter itu sendiri.
2) Algoritma genetika bebas untuk mengkodekan masalah dengan berbagai cara sehingga algoritma genetika tidak dibatasi dengan batasan dari metode lainnya.
3) Algoritma genetika bekerja dengan populasi titik, bukan satu titik.
4) Algoritma genetika menggunakan informasi fungsi tujuan, bukan informasi turunan dan lainnya.
5) Algoritma genetika menggunakan aturan perpindahan probabilistik, bukan deterministik.

6) Algoritma genetika memerlukan iterasi yang berulang-ulang dan dalam jumlah yang relatif banyak, sehingga algoritma ini perlu dibangun dalam sebuah program aplikasi komputer untuk menyelesaikan masalah yang dihadapi.

Kekurangan algoritma genetika

1) Algoritma genetika bekerja dengan bilangan acak pada kromosom awal, sehingga memungkinkan kromosom terbaik tidak terlibat dalam proses.
2) Algoritma genetika menggunakan pembangkitan bilangan random dalam setiap pemilihan kromosom baik untuk induk, proses persilangan maupun mutasi.
3) Solusi yang dihasilkan belum tentu merupakan solusi yang optimal, karena sangat dipengaruhi oleh bilangan acak yang dibangkitkan.

2.1.3 Penerapan Algoritma Genetika

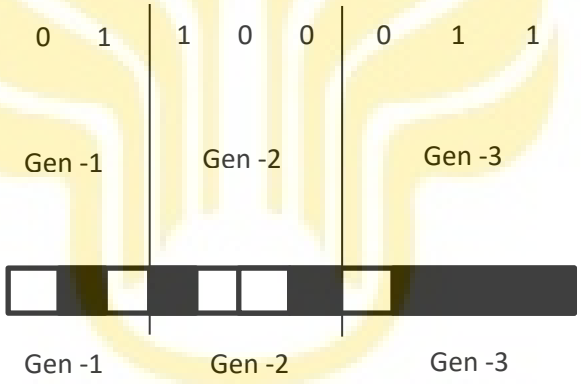
Kusumadewi (2003: 279) menjelaskan bahwa algoritma genetika adalah algoritma pencarian *metaheuristic* yang didasarkan atas mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dari kromosom antar individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidup. Pada dasarnya ada 4 kondisi yang sangat mempengaruhi proses evaluasi, yaitu kemampuan organisme untuk melakukan reproduksi, keberadaan populasi organisme dalam suatu populasi, keberagaman organisme dalam suatu populasi, perbedaan kemampuan untuk *survive*.

Ada 6 komponen dalam algoritma genetika, yaitu:

1) Teknik Penyandian

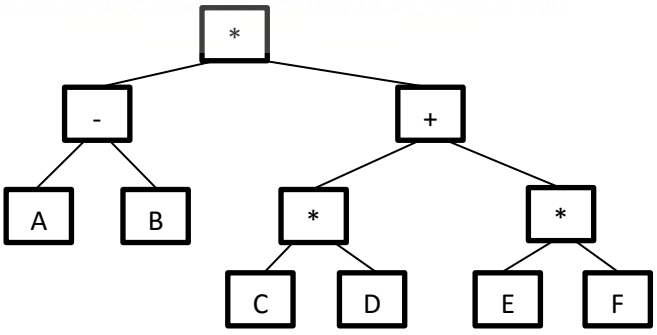
Teknik Penyandian meliputi penyandian gen dari kromosom. Gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variabel.

Gen dapat direpresentasikan dalam bentuk: *string bit*, pohon, *array* bilangan *real*, daftar aturan, elemen permutasi, elemen program, atau representasi lainnya yang dapat diimplementasikan untuk operator genetika. Gambar 2.1 menunjukan representasi *string bit* dan Gambar 2.2 menunjukan representasi pohon.



Gambar 2.1 Representasi *String Bit* Kusumadewi (2003: 281)

(*(-(ab))(+(*(CD))(/(EF))))



Gambar 2.2 Representasi Pohon Kusumadewi (2003: 281)

Demikian juga, kromosom dapat direpresentasikan dengan menggunakan:

a. *String bit* : 10011, 01101, 11101, dst.
b. Bilangan *real* : 65.65, -67.98, 562.88, dst.
c. Elemen permutasi : E2, E10, E5, dst.
d. Daftar aturan : R1, R2, R3, dst.
e. Elemen program : Pemrograman Genetika
f. Struktur lainnya.

2) Prosedur Inisialisasi

Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

3) Fungsi Evaluasi

Ada 2 hal yang harus dilakukan dalam melakukan evaluasi kromosom, yaitu evaluasi fungsi objektif (fungsi tujuan) dan konversi fungsi objektif ke dalam fungsi *fitness*. Secara umum, fungsi *fitness* diturunkan dari fungsi objektif dengan nilai yang tidak negatif. Apabila ternyata fungsi objektif memiliki nilai negatif, maka perlu ditambahkan suatu konstanta C agar nilai *fitness* yang terbentuk menjadi tidak negatif.

4) Seleksi

Seleksi ini bertujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling fit. Ada beberapa metode seleksi dari induk, antara lain *rank-based fitness assignment, roulette wheel selection, stochastic universal sampling, local selection, truncation selection, tournament selection.*

5) Operator Genetika

Ada 2 operator genetika, yaitu:

1) Operator untuk melakukan rekombinasi, yang terdiri dari Rekombinasi bernilai *real* yaitu Rekombinasi diskret, Rekombinasi *intermediate* (menengah), Rekombinasi garis dan Rekombinasi garis yang diperluas. Rekombinasi bernilai biner (*crossover*) yaitu *Crossover* satu titik, *Crossover* banyak titik *Crossover* seragam dan *Crossover* dengan permutasi.
2) Mutasi yang terdiri dari Mutasi bernilai *real* Dan Mutasi bernilai biner.

6) Penentuan Parameter

Yang disebut dengan parameter di sini adalah parameter kontrol algoritma genetika, yaitu ukuran populasi (*popsize*), peluang *crossover* (p_c), dan peluang mutasi (p_m). Nilai parameter ini tentukan juga berdasarkan permasalahan yang akan dipecahkan. Ada beberapa rekomendasi yang bisa digunakan, antara lain:

a) Untuk permasalahan yang memiliki kawasan solusi cukup besar, De Jong merekomendasikan untuk nilai parameter kontrol:

(*popsize*; p_c; p_m) = (50; 0,6; 0,001).

b) Bila rata-rata *fitness* setiap generasi digunakan sebagai indikator =, maka usulannya adalah:

(*popsize*; p_c; p_m) = (80; 0,45; 0.01).

c) Ukuran populasi sebaiknya tidak lebih kecil dari 30, untuk sembarang jenis permasalahan.

Jain (2010: 249) menjelaskan tahapan untuk menyelesaikan sebuah penjadwalan mata kuliah dengan algoritma genetika adalah sebagai berikut.

1) Representasi Kromosom

Kromosom yang akan digunakan untuk mewakili solusi jadwal adalah dengan *array* dua dimensi setiap elemen dari *array* mewakili 3 *tuple* <c,s,r>. Dimana c= representasi kelas, s= mata kuliah yang diambil, r= ruang, n= jumlah periode dalam 1 minggu, k= jumlah dosen.

Batasan solusi penjadwalan mata kuliah yang perlu diperhatikan adalah: (a) Masing-masing dosen tidak dapat ditugaskan lebih dari satu mata kuliah dalam satu periode karena dalam setiap sel *array* tidak boleh lebih dari satu unsur; (b) Kelas, mata kuliah dan ruang tidak boleh dipakai dalam satu periode yang sama.

2) Evaluasi

Evaluasi dilakukan dengan menghitung nilai *fitness*, nilai *fitness* dihitung dari banyaknya pelanggaran kendala yang terjadi setelah proses penyelesaian penjadwalan, dimana untuk menghitung nilai *fitness* digunakan rumus sebagai berikut.

Nilai *Fitness* = $\sum_{i=1}^{\alpha} P(i)$,

Dimana α adalah nilai untuk sebuah pelanggaran.

Metode seleksi algoritma genetika yang digunakan untuk menghasilkan penjadwalan mata kuliah adalah metode *roulette wheel method*. Rumus yang digunakan adalah sebagai berikut.

$$P_i = f_i / \sum_{i=1}^{n} f_i$$

Melalui seleksi kita dapat memilih jadwal yang terbaik dari jadwal yang tersedia. Fungsi *fitness* ini digunakan sebagai ukuran untuk pemilihan kromosom (jadwal).

3) Crossover

Operator *crossover* digunakan untuk menggabungkan dua *string* untuk mendapatkan *string* yang lebih baik. *Crossover* yang digunakan adalah *One site crossover*. Melalui *crossover* diharapkan akan didapatkan jadwal yang optimal dan efektif.

4) Mutasi

Mutasi menambahkan informasi baru dengan cara acak ke proses pencarian genetik dan akhirnya membantu untuk menghindari terjebak di optimal lokal. Operator ini memperkenalkan keragaman dalam populasi setiap kali populasi cenderung menjadi homogen karena penggunaan reproduksi operator *crossover* secara berulang.

Contoh kasus

Terdapat sebuah persamaan a + 2b + 3c = 30, cari nilai a, b dan c yang memenuhi persamaan diatas dengan algoritma genetika.

Penyelesaian:

1) Pembentukan Kromosom

Misal: Batasan nilai variabel a = integer 0 s/d 30.

Sedangkan batasan nilai variabel b dan c adalah bilangan integer 0 s/d 10.

2) Inisialisasi

Misalkan kita tentukan jumlah populasi adalah 3, maka:

Kromosom[1] = [a;b;c] = [12;05;08]

Kromosom[2] = [a;b;c] = [02;06;07]

Kromosom[3] = [a;b;c] = [10;09;03]

3) Evaluasi Kromosom

hitung fungsi_objektif dari kromosom yang telah dibangkitkan:

fungsi_objektif (Kromosom[1]) = Abs((12 + 2*5 + 3*8) – 30)

= Abs((12 + 10 + 24) – 30) = Abs(48 – 30) = 18

fungsi_objektif (Kromosom[2]) = Abs((2 + 2*6 + 3*7) – 30)

= Abs((2 + 12 + 21) – 30) = Abs(35 – 30) = 5

fungsi_objektif (Kromosom [3]) = Abs((10 + 2*9 + 3*3) – 30)

= Abs((10 + 18 + 9) – 30) = Abs(37 – 30) = 7

Rata-rata dari fungsi objektif adalah: (18+5+7)/3 = 10

4) Seleksi Kromosom

fitness[1] = 1 / (fungsi_objektif[1]+1) = 1 / 19 = 0.0526

fitness[2] = 1 / (fungsi_objektif[2]+1) = 1 / 6 = 0.1666

fitness[3] = 1 / (fungsi_objektif[3]+1) = 1 / 8 = 0.125

total_*fitness* = 0.0526 + 0.1666 + 0.125 = 0.4166

Rumus untuk mencari probabilitas: P[i] = *fitness*[i] / total_*fitness*

P[1] = 0.0526 / 0.4166 = 0.1262

P[2] = 0.1666 / 0.4166 = 0.3999

P[3] = 0.125 / 0.4166 = 0.3000

Untuk proses seleksi akan digunakan *roulette wheel*, untuk itu harus dicari dahulu nilai kumulatif probabilitasnya.

C[1] = 0.1262

C[2] = 0.1262 + 0.3999 = 0.5261

C[3] = 0.1262 + 0.3999 + 0.3000 = 0.8261

Kita putar *roulette wheel* sebanyak jumlah populasi yaitu 3 kali (bangkitkan bilangan acak R) dan pada tiap putaran, kita pilih satu kromosom untuk populasi baru. Misal:

R[1] = 0.201

R[2] = 0.284

R[3] = 0.009

Angka acak pertama R[1] adalah lebih besar dari C[1] dan lebih kecil daripada C[2] maka pilih Kromosom[2] sebagai kromosom pada populasi baru, dari bilangan acak yang telah dibangkitkan diatas maka populasi kromosom baru hasil proses seleksi adalah:

Kromosom[1] = Kromosom[2]

Kromosom[2] = Kromosom[2]

Kromosom[3] = Kromosom[1]

Kromosom baru hasil proses seleksi:

Kromosom[1] = [12;05;08]

Kromosom[2] = [02;06;07]

Kromosom[3] = [12;05;08]

5) Crossover

Misal kita tentukan *crossover probability* adalah sebesar 25%, maka diharapkan dalam satu generasi ada 50% kromosom (3 kromosom) dari satu generasi mengalami proses *crossover*. Prosesnya adalah sebagai berikut.

Pertama kita bangkitkan bilangan acak R sebanyak jumlah populasi

R[1] = 0.191, R[2] = 0.259, R[3] = 0.760

Maka kromosom ke k akan dipilih sebagai induk jika R[k] < ρc, dari bilangan acak R diatas maka yang dijadikan induk adalah Kromosom[1] dan Kromosom[2]. Misalkan didapatkan posisi *crossover* adalah 1 maka kromosom induk akan dipotong mulai gen ke 1 kemudian potongan gen tersebut saling ditukarkan antar induk.

Kromosom[1] >< Kromosom[2]

Kromosom[2] >< Kromosom[1]

Posisi *cut-point crossover* dipilih menggunakan bilangan acak 1-2 sebanyak jumlah *crossover* yang terjadi, misal

C[1] = 1

C[2] = 2

offspring[1] = Kromosom[1] >< Kromosom[2]

= [12;05;08] >< [02;06;07] = [12;05;08]

offspring[2] = Kromosom[2] >< Kromosom[1]

= [02;06;07] >< [12;05;08] = [12;05;08]

Dengan demikian populasi kromosom setelah mengalami proses *crossover* menjadi:

Kromosom[1] = [12;05;08]

Kromosom[2] = [12;05;08]

Kromosom[3] = [10;09;03]

6) Mutasi

Misalkan bilangan acak yang terbangkitkan adalah 1 dan 3. Maka populasi kromosom setelah mengalami proses mutasi adalah:

Kromosom[1] = [08;05;12]

Kromosom[2] = [08;05;12]

Kromosom[3] = [03;09;10]

Setelah proses mutasi maka kita telah menyelesaikan satu iterasi dalam algoritma genetika atau disebut dengan satu generasi. Maka fungsi_*objective* setelah satu generasi adalah:

Kromosom[1] = [08;05;12]

fungsi_objektif[1] = Abs((8 + 2*5 + 3*12) – 30)

= Abs((8 + 10 + 36) – 30) = Abs(54 – 30) = 14

Kromosom[2] = [08;05;12]

fungsi_*objektif*[1] = Abs((8 + 2*5 + 3*12) – 30)

= Abs((8 + 10 + 36) – 30) = Abs(54 – 30) = 14

Kromosom[3] = [03;09;10]

fungsi_*objektif*[1] = Abs((3 + 2*9 + 3*10) – 30)

= Abs((3 + 18 + 30) – 30) = Abs(51 – 30) = 21

Rata-rata fungsi objektif setelah satu generasi adalah:

rata-rata = (14 + 14 + 21) / 3 = 39 / 3 = 13

Maka pada generasi selanjutnya kromosom – kromosom yang baru adalah:

Kromosom[1] = [08;05;12]

Kromosom[2] = [08;05;12]

Kromosom[3] = [03;09;10]

Kromosom–kromosom ini akan mengalami proses yang sama seperti generasi sebelumnya yaitu proses evaluasi, seleksi, *crossover* dan mutasi yang kemudian akan menghasilkan kromosom kromosom untuk generasi yang selanjutnya. Proses ini akan berulang sampai sejumlah generasi yang telah ditetapkan sebelumnya.

2.2 Algoritma *Ant Colony Optimization*

2.2.1 Definisi Algoritma *Ant Colony Optimization*

Algoritma *ant colony optimization* (ACO) merupakan sebuah algoritma *metaheuristic*. Algoritma *metaheuristic* merupakan algoritma yang mempunyai kemampuan untuk menangani masalah kompleks *non-linear*, dapat menangani masalah variabel diskrit, dan dapat menangani sebuah masalah optimasi multi-tujuan (Reddy & Bijw, 2016:288).

Dorigo (2001: 12) mendefinisikan bahwa algoritma *ant colony optimization* (ACO) atau algoritma semut adalah algoritma yang awalnya terinspirasi dari perilaku semut, banyak dikembangkan bahwa spesies semut sangat peka terhadap

suara, dan kebanyakan dari spesies mereka benar-benar buta, kebanyakan spesies semut hanya berkomunikasi antara individu dan individu, atau antara individu dan lingkungannya, spesies semut dapat menghasilkan sebuah cairan kimia yang disebut *pheromone*, dimana dengan cairan *pheromone* semut dapat meninggalkan jejak agar jejak tersebut dapat diikuti oleh semut lainnya seperti saat proses pencarian makanan oleh semut.

Pada *ant colony optimization*, sebuah koloni semut buatan bekerjasama untuk menemukan solusi terbaik terhadap sebuah permasalahan optimasi. Semut buatan mempunyai dua sifat (1) mencontoh dari sifat semut yang sesungguhnya yakni menemukan jalan tersingkat oleh koloni semut sungguhan, (2) memperkaya sebuah kemampuan yang tidak terdapat pada semut sungguhan (Dorigo dkk., 2006).

2.2.2 Kelebihan Algoritma *Ant Colony Optimization*

Algoritma *ant colony optimization* mempunyai kelebihan yang tidak dimiliki dari algoritma optimasi lainnya, yaitu:

1) Algoritma semut menggunakan metode umpan balik yang baik sehingga dapat mencapai solusi yang terbaik.
2) Algoritma semut mempunyai sistem kerjasama yang baik. Ditunjukan dengan tingkat keefektifan sebuah pencarian yang melibatkan kerjasama antara koloni semut dalam penyelesaian pada solusi terbaik.
3) Penggunaan struktur yang lebih luas dalam algoritma semut membantu dalam menemukan solusi yang dapat diterima pada tahap proses penelitian.

4) Penggunaan algoritma ini dapat diaplikasikan pada versi yang sama untuk masalah kombinasi optimasi yang berbeda.
5) Algoritma dapat diaplikasikan dalam menyelesaikan masalah kombinasi optimalisasi yang lain, seperti QAP (*Quadratic Assingment Problem*) dan JSP (*Job Shop Scheduling Problem*) dengan perubahan yang tidak terlalu banyak.

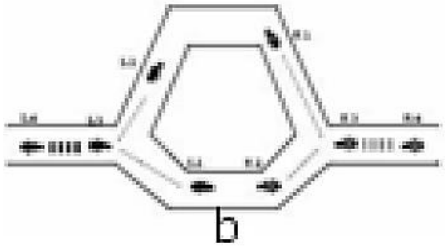
2.2.3 Penerapan Algoritma *Ant Colony Optimization*

Fernandez (2011: 2) mendefinisikan cara kerja algoritma *ant colony optimization* adalah sebagai berikut:

1) Pada awalnya, semut berkeliling secara acak.
2) Ketika semut-semut menemukan jalur yang berbeda misalnya sampai pada persimpangan, mereka akan mulai menentukan arah jalan secara acak seperti pada Gambar 2.3 tingkah laku semut a.

Gambar 2.3 Tingkah Laku Semut a (Fernandez, 2011: 2)

3) Sebagian semut memilih berjalan ke atas dan sebagian lagi akan memilih berjalan ke bawah seperti pada Gambar 2.4 tingkah laku semut b.

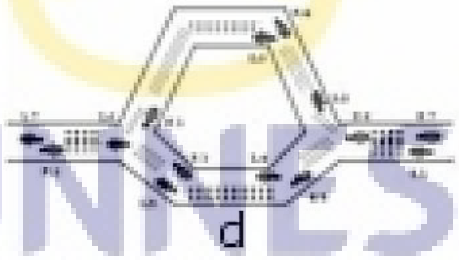


Gambar 2.4 Tingkah Laku Semut b (Fernandez, 2011: 2)

4) Ketika menemukan makanan mereka kembali ke koloninya sambil memberikan tanda dengan jejak *pheromone.*
5) Karena jalur yang ditempuh lewat jalur bawah lebih pendek, maka semut yang bawah akan tiba lebih dulu dengan asumsi kecepatan semua semut adalah sama seperti pada Gambar 2.5 tingkah laku semut c.

Gambar 2.5 Tingkah Laku Semut c (Fernandez, 2011: 2)

6) *Pheromone* yang ditinggalkan oleh semut di jalur yang lebih pendek aromanya akan lebih kuat dibandingkan *pheromone* di jalur yang lebih panjang seperti pada Gambar 2.6 tingkah laku semut d.



Gambar 2.6 Tingkah Laku Semut d (Fernandez, 2011: 2)

7) Semut-semut lain akan lebih tertarik mengikuti jalur bawah karena aroma *pheromone* lebih kuat.

Berlianty & Arifin (2010: 78) mendefinisikan algoritma semut terperinci sebagai berikut:

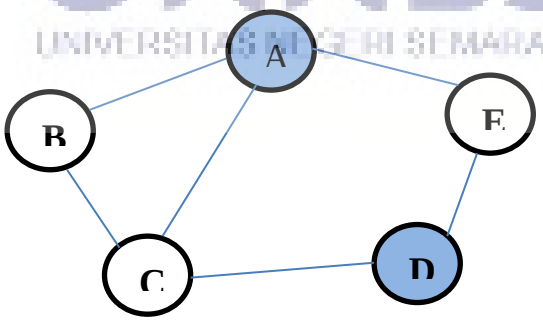
1) Inisialisasi parameter-parameter algoritma, seperti m (jumlah semut), NC max (jumlah siklus maksimum), Q (tetapan siklus semut), α (tetapan pengendali

intensitas jejak semut), β (tetapan pengendali *visibilitas*), ρ (tetapan penguapan jejak semut), dan τ_{ij} (intensitas jejak semut).

2) Inisialisasi *node* pertama setiap semut.
3) Mengisi panjang *node* pertama ke dalam *tabu list*.
4) Menyusun rute kunjungan setiap semut ke setiap *node*.
5) Menghitung panjang rute setiap semut.
6) Mencari solusi terbaik.
7) Menghitung perubahan harga intensitas jejak kaki semut antara *node.*
8) Menghitung harga intensitas jejak kaki semut antara *node* untuk siklus berikutnya.
9) Reset harga perubahan intensitas jejak kaki semut anatar *node*.
10) Mengosongkan *tabu list*.
11) Mengulangi langkah 2 jika diperlukan.

Contoh kasus

Jika diketahui suatu graf di bawah yang ingin diketahui jalur terpendek dari kota A ke kota E, Perhatikan Gambar 2.6 ilustrasi graf dengan 5 kota.



Gambar 2.7 Ilustrasi Graf dengan 5 Kota

Dengan jarak antar kota ($1/d_{ij}$) dapat dilihat pada Tabel 2.1.

Tabel 2.1 Jarak Antar Kota

	A	**B**	**C**	**D**	**E**
A	0	5	7	3	0
B	5	0	4	0	0
C	7	4	0	0	5
D	3	0	0	0	4
E	0	0	5	4	0

Parameter–parameter yang digunakan adalah:

α = 1.00, β = 1.00, ρ = 0.50, τ_{ij} (awal) = 0.01, Maksimum siklus ($NC\ max$) = 2, Tetapan siklus semut (Q) = 1, Banyak semut (m) = 4. Dari jarak kota yang telah diketahui dapat dihitung visibilitas antar kota (η_{ij}) = $1/d_{ij}$. Visibilitas antar kota dapat dilihat pada Tabel 2.2.

Tabel 2.2 Visibilitas Antar Kota

	A	**B**	**C**	**D**	**E**
A	0	0.2	0.143	0.33	0
B	0.2	0	0.25	0	0
C	0.1	0.25	0	0	0.2
D	0.3	0	0	0	0.25
E	0	0	0.2	0.25	0

Siklus ke-1:

Hasil panjang jalur semut pada siklus pertama dapat dilihat pada Tabel 2.3.

Tabel 2.3 Hasil Pada Siklus Pertama

Semut ke-	Rute				Panjang rute
1	A	C	B	E	11
2	A	C	E	-	12
3	A	B	C	E	14
4	A	D	E	-	7

Siklus ke-2:

Hasil panjang jalur semut pada siklus kedua dapat dilihat pada Tabel 2.4.

Tabel 2.4 Hasil Pada Siklus Kedua

Semut ke-	Rute				Panjang rute
1	A	B	C	E	14
2	A	B	C	E	14
3	A	B	C	E	14
4	A	D	E	-	7

Dari dua siklus diatas, diketahui lintasan terpendek diperoleh oleh semut ke-4 dengan jalur A-D-E dengan panjang lintasan 7.

2.3 Penjadwalan

Penjadwalan adalah sebuah pertemuan antara sumber daya dan ruang pada alokasi waktu yang tepat untuk mencapai kelancaran pada tugas sebuah organisasi (Heizer & Render, 1996: 3).

Dalam kehidupan sehari hari penjadwalan banyak digunakan untuk menyelesaikan permasalahan dalam sebuah organisasi seperti penjadwalan di perusahaan industri, penjadwalan di lembaga pendidikan, penjadwalan perhotelan, penjadwalan keberangkatan kereta api, dan sebagainya.

Salah satu penjadwalan yang mempunyai permasalahan yang rumit adalah penjadwalan di tingkat lembaga pendidikan yaitu penjadwalan mata kuliah. Penjadwalan mata kuliah merupakan masalah umum di tingkat universitas di setiap pergantian semester baru.

Lewis (2008: 169) mendefinisikan bahwa terdapat dua kategori permasalahan penjadwalan di universitas yaitu masalah penjadwalan ujian dan

masalah penjadwalan mata kuliah. Kedua jenis masalah tersebut memiliki karakteristik yang hampir sama, perbedaan utamanya adalah dalam penjadwalan ujian, ujian dapat dijadwalkan di ruang yang sama dan pada waktu yang sama (kendalanya adalah tidak melebihi kapasitas duduk), sementara pada penjadwalan mata kuliah, umumnya hanya diperbolehkan setiap perkuliahan per ruang dan per *timeslot*.

2.3.1 Gambaran Umum Penjadwalan Mata Kuliah di Jurusan Ilmu Komputer

Penjadwalan mata kuliah di jurusan ilmu komputer mengikuti aturan yang ditetapkan oleh universitas, dimana ada suatu perkuliahan diampu oleh satu dosen pengampu dan ada perkuliahan lain yang diampu oleh dua dosen pengampu. Perkuliahan aktif dilakukan selama lima hari dalam satu minggu yaitu pada hari senin sampai jumat. Lama perkuliahan ditentukan oleh sks, satu sks sama dengan 50 menit. Pembuatan jadwal perkuliahan dilakukan setelah pengisian KRS atau setelah mahasiswa selesai melakukan pemesanan mata kuliah. Dalam satu angkatan umumnya dibagi menjadi beberapa kelas dalam melakukan perkuliahan pada mata kuliah tertentu, pembagian kelas tersebut tergantung dari kebijakan jurusan. Setiap perkuliahan umumnya dilaksanakan 15 kali pertemuan aktif dan 1 kali digunakan untuk ujian pada satu semester.

2.4 Penelitian Terkait

Penelitian yang terkait digunakan untuk referensi agar dikembangkan oleh peneliti selanjutnya. referensi terkait mempunyai keterkaitan metode dan objek penelitian terhadap penelitian yang akan dilakukan. Berikut beberapa penelitian yang terkait dengan penelitian yang akan dibuat:

1) Mahiba & Durai (2012: 253) dalam jurnalnya yang berjudul "*Genetic Algorithm with Search Bank Strategis for University Course Timetabling Problem*". Menjelaskan tentang usulan sebuah metode baru algoritma genetika dengan perpaduan *Search Bank Strategi local, guided* dan *tabu search*. *Local Search* digunakan untuk meningkatkan keturunan atau solusi. *Guided Search* digunakan untuk mempersempit solusi dengan menggunakan *Events* Struktur Data. *Tabu search* digunakan untuk menghapus solusi yang digunakan. Metode baru yang diusulkan memberikan hasil yang menjanjikan untuk UCTP (*University Course Timetabling Problem*).
2) Arifudin (2012: 1) dalam jurnalnya yang berjudul "Optimasi Penjadwalan Proyek dengan Penyeimbangan Biaya Menggunakan Kombinasi CPM dan Algoritma Genetika". Melakukan penelitian dengan mengkombinasikan CPM dengan algoritma genetika untuk melakukan penjadwalan sebuah proyek. Alokasi kegiatan ditentukan berdasarkan waktu mulai paling awal dan waktu mulai terakhir dengan memperhitungkan biaya dalam sumber daya setiap periode proyek. Penjadwalan proyek dan kriteria optimal digunakan untuk meminimalkan biaya penyimpangan dari rata-rata total biaya proyek. Hasil yang diperoleh dalam penelitian ini dengan kombinasi CPM dan algoritma

genetika dapat menghasilkan jadwal proyek lebih cepat dan biaya proyek per hari juga lebih hemat. Metode penjadwalan ini dapat menjadi alternatif keputusan untuk kontraktor dalam pelaksanaan proyek.

3) Susiloputro (2012: 1) dalam jurnalnya yang berjudul "Penerapan Pewarnaan Graf Pada Penjadwalan Ujian Menggunakan Algoritma *Welsh Powell*". Menjelaskan tentang pembuatan jadwal ujian akhir dengan menggunakan algoritma *Welsh Powell* yang telah dimodifikasi penyusunan jadwal diawali dengan membuat model graf konflik penjadwalan menggunakan data peserta kuliah. Kemudian dilakukan pewarnaan pada graf konflik penjadwalan menggunakan algoritma *Welsh Powell* yang telah dimodifikasi. Setelah diperoleh hasil pewarnaan maka dapat disusun jadwal ujian akhir semester berdasarkan hasil pewarnaan. Mata kuliah dengan warna sama dapat dijadwalkan pada waktu yang bersamaan dan sebaliknya.
4) Badoni dkk., (2014: 12) dalam jurnalnya yang berjudul "*A New Hybrid Algorithm for University Course Timetabling Problem using Events Based on Groupings of Students*". Menjelaskan tentang algoritma *hybrid* baru (NHA) menggabungkan algoritma genetika dengan *local search* dan menggunakan aktivitas berdasarkan pengelompokan siswa untuk memecahkan masalah penjadwalan di universitas. Sebuah daftar aktivitas seperti perkuliahan, pemberian tutorial, penggunaan laboratorium dan seminar yang dilakukan dan kelompok siswa yang saling beririsan dalam mengambil daftar maka sekali seorang mahasiswa dipilih dalam setiap kelompok, ia dikeluarkan dari seleksi selanjutnya dalam kelompok-kelompok lainnya. Gabungan dari semua

aktivitas yang diambil oleh semua siswa dari masing-masing kelompok terbentuk. Jumlah aktivitas dalam setiap kelompok disebut sebagai ukuran kelompok yang terikat dibatasi dengan jumlah total timeslots dan dapat dikurangi dengan jumlah maksimum aktivitas per siswa. Proses kelompok di atas yang terbentuk diulang sampai ukuran masing-masing kelompok berkurang dalam hal ini terikat dengan tidak memilih aktivitas-aktivitas yang umum untuk semua siswa dalam kelompok. Sekarang, algoritma genetika dengan *local search* (GALS) diterapkan pada sejumlah tolak ukur masalah. Hasil percobaan menunjukkan bahwa algoritma kami, NHA, mampu menghasilkan hasil yang menjanjikan jika dibandingkan dengan hasil yang diperoleh dengan menggunakan GALS dan algoritma yang ada.

5) Shiau (2011: 235) dalam jurnalnya yang berjudul "*A hybrid particle swarm optimization for a university course scheduling problem with flexible preferences*". Mengusulkan tentang sebuah algoritma *metaheuristic* baru yang didasarkan pada prinsip-prinsip particle *swarm optimization* (PSO) yang diusulkan untuk masalah penjadwalan kuliah. Algoritma mencakup beberapa fitur: merancang sebuah 'nilai posisi mutlak' representasi untuk partikel; memungkinkan pengajar bahwa mereka bersedia untuk kuliah berdasarkan keinginan mereka, seperti hari yang mereka sukai dan periode waktu, jumlah maksimum periode waktu mengajar bebas dan format perkuliahan (periode waktu berturut-turut atau dipisahkan menjadi periode waktu yang berbeda), dan mempekerjakan proses perbaikan untuk semua jadwal yang tidak layak. Algoritma diuji dengan menggunakan data penjadwalan dari universitas di

Taiwan. Hasil percobaan menunjukkan bahwa algoritma *hybrid* yang diusulkan menghasilkan solusi yang efisien dengan hasil yang optimal tentu saja penjadwalan untuk pengajar dan pengaturan penjadwalan kelas. Algoritma *hybrid* ini juga melebihi algoritma genetika yang diusulkan dalam literatur.

6) Pongcharoen dkk., (2008: 903) dalam jurnalnya yang berjudul "*Stochastic Optimisation Timetabling Tool for university course scheduling*". Menjelaskan tentang *Stochastic Optimization* Penjadwalan *Timetabling Tool* (SOTT) yang telah dikembangkan untuk penjadwalan di universitas. Algoritma Genetika (GA), *Simulated Annealing* (SA) dan pencarian acak yang tertanam di SOTT tersebut. Algoritma termasuk proses perbaikan, yang menjamin bahwa semua jadwal tidak layak akan diperbaiki. Hal ini untuk mencegah bentrokan dan memastikan bahwa ruang yang besar cukup untuk menampung kelas. Algoritma juga mengevaluasi jadwal dalam hal batasan yaitu: meminimalkan protes dari mahasiswa; menghindari fragmentasi dalam jadwal untuk mahasiswa dan dosen; dan memuaskan keinginan dosen dalam waktu perkuliahan. Algoritma diuji dengan menggunakan dua set data penjadwalan dari kolaborasi universitas. Kedua GA dan SA memproduksi jadwal yang sangat baik, tapi hasil yang diperoleh dari SA sedikit lebih baik dari GA. Namun, GA mendapatkan 54% lebih cepat dari SA.
7) Al Salami (2009: 824) dalam jurnalnya yang berjudul "*Ant Colony Optimization Algorithm*". Mengusulkan tentang algoritma *hybrid* yang digunakan untuk memecahkan masalah optimasi kombinatorial dengan

menggunakan Algoritma *Ant Colony Optimization* dan Algoritma Genetika. Proses evolusi dari algoritma *Ant Colony Optimization* menyesuaikan operasi genetik untuk meningkatkan gerakan *ant* terhadap keadaan solusi. Algoritma konvergen terhadap solusi akhir yang optimal, dengan mengumpulkan sub-solusi paling efektif.

BAB V

SIMPULAN DAN SARAN

5.1 Simpulan

Dari percobaan yang dilakukan dalam penelitian didapat performansi terbaik algoritma genetika yaitu dengan waktu eksekusi: 21,26 *Second* dan memori yang digunakan: 12.159,08 *Kilo byte*. Sedangkan percobaan yang dilakukan terhadap algoritma *ant colony optimization* mendapatkan performansi terbaik dengan waktu eksekusi: 69,11 *Second* dan memori yang digunakan: 21.674,48 *Kilo byte.* Berdasarkan hasil performansi terbaik dari kedua algoritma dalam menyelesaikan penjadwalan mata kuliah di jurusan ilmu komputer, dapat ditarik kesimpulan bahwa algoritma genetika mempunyai performansi yang lebih baik dari algoritma *ant colony optimization*. Algoritma genetika mendapatkan solusi terbaik lebih cepat dan lebih sedikit menggunakan memori dibandingkan dengan algoritma *ant colony optimization* saat melakukan proses komputasi.

5.2 Saran

Beberapa saran yang dapat digunakan dalam penelitian ini adalah sebagai berikut:

1. Diharapkan ada penelitian lanjut yang menggunakan algoritma *metaheuristic* lainnya untuk menyelesaikan kasus penjadwalan mata kuliah untuk mengetahui performansi algoritma mana yang paling baik untuk digunakan.

2. Diharapkan dipenelitian mendatang akan ada kombinasi antara algoritma yang digunakan dalam penelitan dengan algoritma *metaheuristic* lainnya untuk mendapatkan performansi yang lebih baik.

DAFTAR PUSTAKA

Alamsyah, Wardoyo, R. 2004. Optimalisasi Penjadwalan Multi Constraint Menggunakan Logika Fuzzy = Multi Constraint Scheduling Optimization Using Fuzzy Logic. *Sains dan Sibernatika*, 17.

Al Salami, N. M. A. 2009. Ant colony optimization algorithm. *UbiCC Journal*, 4(3), 823-826.

Arifudin, R. 2012. Optimasi Penjadwalan Proyek dengan Penyeimbangan Biaya menggunakan Kombinasi CPM dan Algoritma Genetika. *Jurnal Masyarakat Informatika*, 2(4): 114.

Babaei, H., Karimpour, J., & Hadidi, A. 2015. A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, 86: 43-59.

Badoni, R. P., Gupta, D. K., & Mishra, P. 2014. A New Hybrid Algorithm for University Course Timetabling Problem using Events Based on Groupings of Students. *Computers & Industrial Engineering*, 78: 12-25.

Berlianty, I., & Miftahol, A. 2010. *Teknik Teknik Optimasi Heuristik*. Graha Ilmu. Yogyakarta.

Dorigo, M. 2001. *Ant Algorithms Solve Difficult Optimization Problems*. In Advances in Artificial Life. Springer Berlin Heidelberg.

Dorigo, M., Birattari, M., & Stützle, T. 2006. Ant Colony Optimization. *Computational Intelligence Magazine*, IEEE, 1(4): 28-39.

Fernandez, A., Handoyo, E., & Somantri, M. 2011. Pembangunan Aplikasi Penyusunan Jadwal Kuliah Menggunakan Algoritma Semut (*Doctoral dissertation, Jurusan Teknik Elektro Fakultas Teknik*).

Heizer, J., & Render, B. 1996. *Production and operations management (4th ed.)*. Upper Saddle River, NJ: PrenticeHall.

Hong, S. S., Lee, W., & Han, M. M. 2015. The Feature Selection Method based on Genetic Algorithm for Efficient of Text Clustering and Text Classification. *International Journal of Advances in Soft Computing & Its Applications*, 7(1).

Jain, A., Jain, S., & Chande, P. K. 2010. Formulation of Genetic Algorithm to Generate Good Quality Course Timetable. *International Journal of Innovation, Management and Technology*, 1(3): 248.

Jogiyanto. 2008. *Metodologi Penelitian Sistem Informasi*. Yogyakarta: Andi.

Kusumadewi, S. 2003. *Artifical Intelligence*. Yogyakarta: Graha Ilmu.

Ladjamudin, A. B. B. 2006. *Rekayasa Perangkat Lunak*. Graha Ilmu, Yogyakarta.

Lewis, R. 2008. A survey of metaheuristic-based techniques for university timetabling problems. *OR spectrum*, 30(1), 167-190.

Mahiba, A. A., & Durai, C. A. D. 2012. Genetic algorithm with search bank strategies for university course timetabling problem. *Procedia Engineering*, 38, 253-263.

Nugraha, D., & Kosala, R. 2014. A Comparative Study Of Evolutionary Algorithms For School Scheduling Problem. *Journal Of Theoretical & Applied Information Technology*, 67(3).

Pongcharoen, P., Promtet, W., Yenradee, P., & Hicks, C. 2008. Stochastic Optimisation Timetabling Tool for University Course Scheduling. *International Journal of Production Economics*, 112(2): 903-918.

Pressman, Roger S. 2002. *Rekayasa Perangkat Lunak*. Yogyakarta: Andi.

Reddy, S. S., & Bijwe, P. R. 2016. Efficiency improvements in metaheuristic algorithms to solve the optimal power flow problem. *International Journal of Electrical Power & Energy Systems*, 82, 288-302

Saragih, H., Hoendarto, G., Reza, B., & Setiyadi, D. 2012. Aplikasi Sistem Perangkat Lunak Menggunakan Algoritma Ant untuk Mengatur Penjadwalan Kuliah. *Teknik dan Ilmu Komputer*, 1(3): 256-241.

Shiau, D. F. 2011. A Hybrid Particle Swarm Optimization for a University Course Scheduling Problem with Flexible Preferences. *Expert Systems with Applications*, 38(1): 235-248.

Singh, S., Dubey, G. C., & Shrivastava, R. Ant Colony Optimization Using Genetic Algorithms. *International Journal of Theoretical and Applied Science*, 4(1): 48-51.

Sitarz, P., & Powałka, B. 2016. Modal parameters estimation using ant colony optimisation algorithm. *Mechanical Systems and Signal Processing*, 76, 531-554.

Sugiyono. 2013. *Metode Penelitian Kuantitatif, Kualitatif dan R&D*. Bandung: Alfabeta.

Susiloputro, A., Rochmad, R., & Alamsyah, A. 2012. Penerapan Pewarnaan Graf pada Penjadwalan Ujian menggunakan Algoritmawelsh Powell. *Unnes Journal of Mathematics*, 1(1).

Squillero, G., & Tonda, A. 2016. Divergence of Character and Premature Convergence: A Survey of Methodologies for Promoting Diversity in Evolutionary Optimization. *Information Sciences*, 329: 782-799.

Tiwari, P. K., & Vidyarthi, D. P. 2016. Improved Auto Control Ant Colony Optimization using Lazy Ant Approach for Grid Scheduling Problem. *Future Generation Computer Systems*, 60: 78-89.