Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi

Vol. 5, No. 1, Januari 2009 ISSN 0216-0544

DAFTAR ISI

IMPLEMENTASI PENJADWALAN MATAKULIAH MENGGUNAKAN ALGORITMA KOLONI SEMUT (AKS) DI JURUSAN TEKNIK INFORMATIKA ITS **1-11**
Sarwosri, Devie Rosa Anamisa

APLIKASI BASIS DATA *FUZZY* BERBASIS *WEB* UNTUK PEMILIHAN *HANDPHONE* **12-23**
Shofwatul 'Uyun

PEMISAHAN SUMBER-SUMBER BUNYI MENGGUNAKAN *INDEPENDENT COMPONENT ANALYSIS* **24-31**
Mula'ab

IDENTIFIKASI POHON KELAPA SAWIT PADA CITRA FOTO UDARA MENGGUNAKAN OPERASI MORFOLOGI BERBASIS *REGION* **32-39**
Soffiana Agustin

ANALISA PEMANFAATAN FUNGSI PENALTI PADA KOMPUTASI PENYELESAIAN PERMASALAHAN OPTIMASI NONLINIER **40-47**
Victor Hariadi

RANCANG BANGUN SISTEM AGEN CERDAS MONITORING STOK PERUSAHAAN **48-58**
Rika Yunitarini

PENGGUNAAN *CLUSTER-BASED SAMPLING* UNTUK PENGGALIAN KAIDAH ASOSIASI MULTI OBYEKTIF **59-67**
Febriana Santi Wahyuni, Daniel O Siahaan dan Chastine Fatichah

SISTEM TEMU KEMBALI CITRA GEDUNG BERDASARKAN INFORMASI GARIS PADA BENTUK GEDUNG **68-74**
Iman Sapuguh, Daniel O Siahaan, dan Chastine Fatichah

Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi

Vol. 5, No. 1, Januari 2009 **ISSN 0216 - 0544**

IMPLEMENTASI PENJADWALAN MATAKULIAH MENGGUNAKAN ALGORITMA KOLONI SEMUT (AKS) DI JURUSAN TEKNIK INFORMATIKA ITS

[*] Sarwosri, [**] Devie Rosa Anamisa
Jurusan Teknik Informatika, Fakultas Teknologi Informasi Institut Teknologi Sepuluh Nopember
Gedung Baru Teknik Informatika, Jl. Teknik Kimia Kampus ITS, Sukolilo, Surabaya
E-Mail : [*] sri@its-sby.edu, [**] devros_gress@yahoo.com

Abstract

In the university, course scheduling is one of important things in learning process. All of the activity of lecturers and students will be depended on the schedule, so it must be fixed in the early of academic year. To solve the course scheduling problems, ant colony algorithm is the acceptable solution, since ant colony algorithm is an agent based algorithm which is able to solve course scheduling problems at informatics department of ITS (Sepuluh Nopember Institute of Technology effectively). Based on the conducted test, this software is able to handle some of scheduling requests in a certain time with existing constraint. Therefore it can produce scheduling application which do not have a collision and is able to give the information about active rooms utility and average of sks (credit point) load of an active lecturer so that the result of scheduling can be used

Key words: *Ant Colony Algorithms, scheduling problem.*

PENDAHULUAN

Pembuatan jadwal matakuliah pada Jurusan Teknik Informatika ITS harus dilakukan pada setiap pergantian semester. Padahal proses pembuatan jadwal ini membutuhkan waktu, tenaga dan ketelitian untuk membuatnya. Karena itu diperlukan suatu penjadwalan otomatis yang dapat membuat jadwal dengan cepat dan mudah sehingga masalah pembuatan jadwal matakuliah dapat diselesaikan dengan lebih efisien.

Dalam pembuatan jadwal matakuliah harus memperhatikan aturan-aturan penjadwalan yang sudah ditentukan dengan memastikan mahasiswa mengambil matakuliah sesuai dengan kurikulum yang ditentukan jurusan. Ada beberapa faktor yang berpengaruh dalam pembentukan jadwal seperti dosen, ruang, waktu matakuliah, jumlah mahasiswa, beban sks minimal dan maksimal seorang dosen dan daya tampung suatu ruang. Semua faktor tersebut ditujukan untuk mendukung dalam masalah penjadwalan matakuliah di Jurusan Teknik Informatika ITS.

Cukup banyak teknik yang telah digunakan untuk membuat penyelesaian permasalahan pembuatan jadwal. Akan tetapi untuk menyelesaikan sebuah permasalahan penjadwalan matakuliah di Jurusan Teknik Informatika ITS, sebuah solusi yang optimum sangat tidak memungkinkan untuk dicari. Karena itulah pencarian solusi yang mendekati optimum sudah dirasakan cukup. Beberapa teknik yang dapat digunakan untuk mendapatkan solusi tersebut adalah dengan menggunakan algoritma berbasis *agent*.

Salah satu algoritma berbasis *agent* yang dapat digunakan untuk menyelesaikan permasalahan optimasi kombinatorial seperti permasalahan penjadwalan matakuliah adalah Algoritma Koloni Semut (AKS). AKS merupakan algoritma yang terinspirasi dari perilaku semut yang mencari makan hingga semut itu kembali ke koloninya dengan memberikan tanda dengan jejak *feromon*. Jejak *feromon* merupakan suatu perantara komunikasi tidak langsung dalam mendapatkan informasi untuk menemukan jalur terpendek, yang dapat membuat semut tidak lagi

berpergian secara acak dalam membangun suatu solusi dengan cepat. Di dalam optimasi kombinatorial, AKS bekerja secara lebih efektif daripada algoritma genetika serta lebih menguntungkan dari pendekatan penguatan tiruan (*simulated annealing*) dan algoritma genetika saat grafik mungkin berubah secara dinamis.

Dalam pembuatan jadwal matakuliah di Jurusan Teknik Informatika ITS, harus memperhatikan aturan-aturan penjadwalan matakuliah yang sudah ditentukan untuk menghindari terjadinya bentrok antar komponen-komponen yang dijadwalkan.

ALGORITMA KOLONI SEMUT

Algoritma Koloni Semut (AKS) adalah metode optimasi yang telah digunakan untuk menyelesaikan aplikasi seperti *traveling salesmen problem*, *vehicle-routing problem*, *quadratic assigment problem*, *multidimensional knapsack problem* dan sebagainya[1].

Koloni semut memiliki tingkah laku yang sangat menarik. Walaupun satu ekor semut memiliki kemampuan yang sangat sederhana, tetapi pola hidup semut-semut tersebut dalam koloninya sangat terstruktur[2][3]. Dengan kemampuan komunikasi yang sangat terbatas, semut yang tidak dapat berinteraksi dengan semut yang lain berdasarkan pada aliran informasi yang sangat sederhana untuk mencari makanan[2].

Media komunikasi yang digunakan oleh semut sebagai aliran informasi adalah jejak *feromon*. Jejak *feromon* yang ditinggalkan oleh masing-masing individu semut ini dapat digunakan untuk mengoptimalkan lintasan dari sarang menuju sumber makanan. AKS ini didasarkan pada kemampuan koloni semut untuk menemukan jalan terpendek dalam mencari sumber makanan dan menghindari rintangan[2].

AKS menggunakan *agent* sederhana disebut dengan *ant* yang secara iteratif menemukan solusi yang optimal dalam permasalahan kombinasi. Algoritma ini dapat diterapkan pada permasalahan kombinasi dengan menentukan komponen solusi yang dapat digunakan oleh *ant* untuk secara iteratif menemukan solusi yang optimal dengan meninggalkan jejak *feromon*.

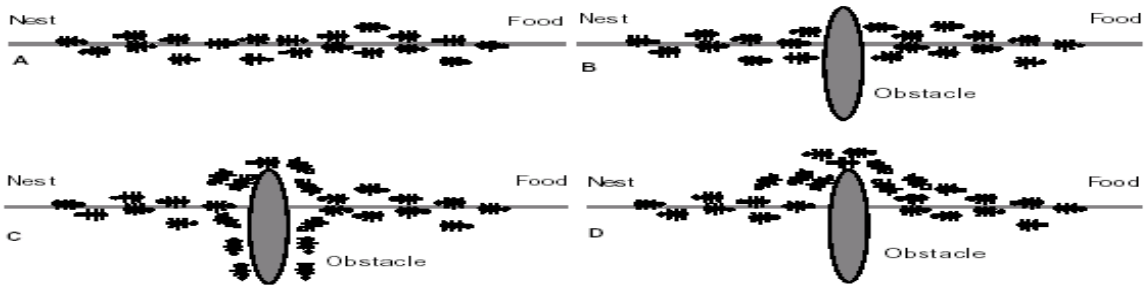
Jejak *feromon* dalam permasalahan ini menyatakan intensitas *ant* dalam mengunjungi atau menggunakan suatu komponen solusi. Setiap *ant* membangun sebuah solusi diawali dengan solusi *null* kemudian komponen solusi akan ditambahkan satu per satu hingga terbentuk sebuah solusi yang lengkap.

Setelah solusi terbentuk, *ant* meninggalkan jejak *feromon*nya pada setiap komponen solusi yang digunakan pada solusi yang terbentuk. Oleh karena itu biasanya komponen solusi yang merupakan bagian dari solusi terbaik atau yang sering digunakan dalam iterasi akan memiliki jejak *feromon* dengan jumlah yang lebih besar. Hal ini mempengaruhi kecenderungan komponen solusi tersebut untuk digunakan oleh *ant* yang lain dalam iterasi selanjutnya.

Proses pada Gambar 1 menunjukkan bahwa setiap semut hanya memberikan kontribusi yang sangat sedikit. Menarik untuk diperhatikan bahwa meskipun seekor semut saja mampu untuk membangun sebuah solusi yang berupa sebuah jalur antara sarang dengan sumber pangan, hanya dengan kerja sama dari semut-semut itulah, maka koloni semut dapat memberikan suatu solusi terbaik berupa jalur terpendek. Juga menarik untuk diperhatikan bahwa semut dapat melakukan hal ini dengan menggunakan sebuah komunikasi sederhana yang diperantarai oleh peletakan bentuk komunikasi tidak langsung yang diperantarai oleh modifikasi terhadap lingkungan.

PENJADWALAN KULIAH DENGAN ALGORITMA KOLONI SEMUT

Penelitian ini membahas bagaimana menyelesaikan permasalahan penjadwalan matakuliah dengan menggunakan AKS. Seperti tingkah laku semut pada umumnya bahwa semut memiliki kemampuan yang sangat sederhana dan pola hidup yang sangat terstruktur[2][3].



Gambar 1. Semut Mengeksploitasi *Feromon* untuk Menemukan *Shortest path* antara Dua Titik [4].

Secara garis besar proses yang dilakukan untuk penyelesaian masalah penjadwalan dengan AKS ditunjukkan dalam *flowchart* pada Gambar 3[2].

Untuk mencapai penjadwalan yang optimal semut harus melakukan beberapa tahap di antaranya:

1. Tahap I: Inisialisasi Parameter.
   Tahap awal dari proses penjadwalan ini adalah inisialisasi parameter. Parameter-parameter yang digunakan dalam proses ini antara lain [2]:
   - *Seed sequence* adalah kumpulan dari matakuliah tawar yang telah ditempatkan pada sejumlah waktu dan ruang yang tersedia dalam penjadwalan dimana penempatan waktu dan ruang tersebut didasarkan pada *constraint* yang ada.
   - α, β merupakan parameter-parameter lain yang mengontrol bobot relatif antara jejak *feromon* dan nilai heuristik. Jika α=0, maka pencarian akan berdasarkan nilai heuristik semata. Jika sebaliknya (β=0), maka pencarian hanya dituntun oleh informasi jejak *feromon*.
   - ρ adalah sebuah koefisien dimana (1-ρ) menunjukkan tingkat evaporasi *feromon*. ρ memiliki nilai yang sama untuk semua iterasi yaitu 0.5.
   - Z_{best} adalah besarnya jumlah data *seed sequence* dalam lima hari aktif kuliah (senin sampai dengan jumat). Nilai Z_{best} akan diperbarui pada setiap iterasi.
   - T_{max} dan T_{min} adalah batas maksimum dan minimum dari nilai intensitas elemen matriks *feromon*. T_{max} dan T_{min} diperoleh dari Z_{best} dengan formula sebagai berikut:
     T_{max}= 1/((1-ρ)Z_{best}) (1)
     T_{min} = T_{max} /5 (2)
     $\tau_{ij}(t)$ adalah intensitas jejak *feromon matakuliah i* diposisi *k*. Pada iterasi awal $\tau_{ij}(t)$ Yaitu T_{max} (3)

Tabel 1 adalah contoh inisialisasi parameter dalam algoritma koloni semut dengan tiga contoh data *seed sequence*.

- *Zbest* yang digunakan adalah besarnya nilai dari *seed sequence* dibagi dengan jumlah hari aktif yaitu 1,2,3,.... 117 dalam 5 hari aktif kuliah. Jadi besarnya adalah 23.
- ρ, α dan β memiliki nilai yang sama untuk semua iterasi yaitu 0.5, 1 dan 2.
- *Tmax* = (1/(1-ρ)*Zbest*)= (1/1-0.5)9)= 0.222.
- *Tmin* = *Tmax*/5= 0.222/5 = 0.044.
- $\tau_{ij}(t)$ = Tmax untuk iterasi pertama. Oleh karena itu $\tau_{ij}(t)$ untuk masing-masing matakuliah *i* di posisi *j* disimpan dalam matriks *n x n* sebagaimana ditunjukkan pada Tabel 2.

Tabel 1. Inisialisasi Parameter.

Id_mk	NIP	Kelas	Ruang	Hari	Jam
CI1411	132297166	X	6	1	2
CI1411	132297166	X	1	1	3
CI1402	132296226	C	2	1	3

Tabel 2. Matriks Nilai.

i/j	1	2	3	4	5	6
1	0.222	0.222	0.222	0.222	0.222	0.222
2	0.222	0.222	0.222	0.222	0.222	0.222
3	0.222	0.222	0.222	0.222	0.222	0.222
4	0.222	0.222	0.222	0.222	0.222	0.222
5	0.222	0.222	0.222	0.222	0.222	0.222
6	0.222	0.222	0.222	0.222	0.222	0.222

2. Tahap II: Iterasi AKS.

Suatu iterasi akan menghasilkan sebuah solusi yaitu jadwal dari urutan sejumlah matakuliah yang ditawarkan. Solusi dari setiap iterasi dioptimalkan lagi dalam iterasi selanjutnya berdasarkan jejak *feromon* yang nilainya berubah-ubah tergantung pada solusi yang dihasilkan pada iterasi sebelumnya.

Suatu solusi dinyatakan lebih optimal jika waktu penyelesaian sejumlah matakuliah yang akan dijadwalkan oleh jadwal yang baru terbentuk besarnya lebih kecil daripada waktu penyelesaian sejumlah matakuliah yang akan dijadwalkan pada iterasi sebelumnya. Iterasi algoritma koloni semut terdiri dari tiga bagian penting yaitu pembentukan solusi, penyempurnaan solusi dan *update* matriks *feromon*.

Pembentukan Solusi

Proses pembentukan solusi dilakukan dengan menambahkan matakuliah satu persatu ke dalam solusi (jadwal) dengan melakukan langkah-langkah berikut:

Untuk memilih matakuliah *i* yang belum terjadwal untuk dijadwalkan pada posisi *k* dalam jadwal adalah sebagai berikut:

1. Tentukan nilai Produk *feromon* yang dikontrol dalam *decision table* $A_i = [a_{ij}(t)]_{|N_i|} : a_{ij}(t) = [\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}$, η_{ij} adalah nilai heuristik untuk berpindah dari *state i* ke *state j* (4)
2. Tentukan nilai *Tik* (akumulasi dari Produk *Feromon*):

   Tij = Σ $a_{ij}(t)$ (5)
3. Hitung nilai *u* dengan *random number generator* dalam banyaknya jumlah semut untuk dibangkitkan.

   u = *Math.random*()**jmlSemut*, dimana jumlah Semut < 10

   Dari random number *generator* ini akan dihasilkan suatu bilangan *random* selanjutnya disebut dengan *u* yang berada dalam range (0,1)
   - *u*<=(jumlah matakuliah yang akan dijadwalkan -*n*)/*n*
     maka matakuliah *i* dipilih berdasarkan nilai *Tik* terbesar dari nilai *u*.
     n adalah jumlah matakuliah dalam matriks.
   - jika tidak maka hitung nilai probabilitasnya

$$p_{ij}^{k} = \frac{a_{ij}(t)}{\sum_{l \in N_i^k} a_{il}(t)} \qquad (6)$$

Pik adalah probabilitas dari matakuliah *i* di posisi *k*. Dan $N_i^k \subseteq N_i$ adalah himpunan tetangga *node i* yang masih harus dikunjungi semut *k*.

Matakuliah yang dipilih adalah matakuliah yang memiliki probabilitas *Pik* yang terbesar.

Setelah semua matakuliah dijadwalkan satu persatu, maka akan terbentuk satu solusi berupa jadwal urutan pengerjaan dari semua matakuliah. Jadwal ini kemudian disempurnakan lagi melalui proses *local search*.

Penyempurnaan Solusi

Proses penyempurnaan solusi dilakukan dengan *local search* menggunakan metode *job-index-based*. Proses *local search* ini dilakukan sebanyak tiga kali. Langkah-langkah proses ini adalah sebagai berikut [2]:

for i=1 to n
for k=1 to i
if Q[i] ≠ k then (Q adalah jadwal yang dihasilkan dari proses iterasi pada proses pembentukan solusi)

Keterangan:

- Masukkan *i* di posisi *k*
- Jika waktu penyelesaian keseluruhan matakuliah yang akan dijadwalkan yang baru lebih baik dari keseluruhan waktu *Q* maka jadwal ini disimpan sebagai solusi yang baru (*Q*).

Setelah *local search* dilakukan sebanyak tiga kali, nilai *Q* yang terakhir adalah solusi terbaik sementara yang diperoleh pada iterasi ini.

Update Matriks *Feromon*

Setelah diperoleh solusi terbaik sementara dari proses *local search* maka dilakukan *update* matriks *feromon* berdasarkan solusi terbaik sementara yang dihasilkan. Langkah-langkah yang dilakukan adalah [2]:

1. Hitung nilai *Zbest*, *Tmax* dan *Tmin* baru berdasarkan solusi terbaik sementara.

2. Untuk setiap $\tau_{ij}(t)$
   - Jika matakuliah *i* dijadwalkan pada posisi *k*, maka
     $\tau_{ij}(t)$ baru=ρ* $\tau_{ij}(t)$ lama+(1/*Zbest*) (7a)
   - Jika tidak, maka
     $\tau_{ij}(t)$ baru=ρ * $\tau_{ij}(t)$ lama (7b)
     Jika nilai $\tau_{ij}(t)$ baru lebih besar dari Tmax maka $\tau_{ij}(t)$ *baru* = *Tmax*. Sedangkan jika nilai $\tau_{ij}(t)$ baru lebih kecil dari *Tmin*, maka $\tau_{ij}(t)$ *baru* = *Tmin*[3].

Setelah proses *update* matriks *feromon* maka dilakukan pengecekan terhadap kondisi perulangan untuk pengecekan *constraint* telah dilanggar atau tidak dengan kondisi perulangannya sebesar yang diinginkan oleh pengguna adalah :

Jumlah iterasi <= besarnya iterasi yang dimasukkan oleh user

Jika kondisi perulangan masih terpenuhi maka dilakukan proses iterasi berikutnya dengan menggunakan parameter-parameter yang dihasilkan pada proses *update* matriks *feromon*. Proses perulangan akan berhenti sesuai dengan keinginan pemakai. Sehingga solusi terbaik sementara akan dihasilkan pada iterasi terakhir dengan semua *constraint* dasar penjadwalan terpenuhi.

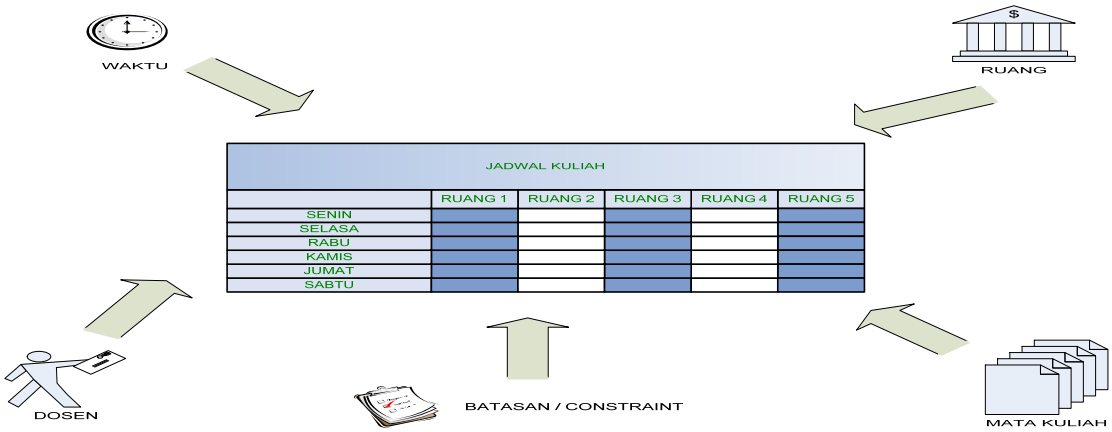
PERANCANGAN SISTEM DAN PERANGKAT LUNAK

Faktor-faktor yang mempengaruhi penjadwalan dapat dilihat pada Gambar 2.

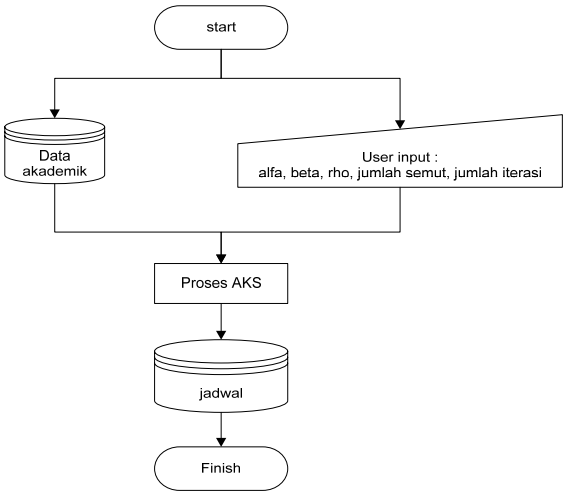
Aplikasi penjadwalan matakuliah ini dipengaruhi oleh beberapa komponen yang terdiri Dosen, Ruang, Waktu, Matakuliah.

Adapun *hard constraint* yang diterapkan dalam penjadwalan matakuliah ini antara lain adalah:

1. Ada sejumlah matakuliah yang ditawarkan untuk dibuat jadwalnya.
2. Setiap matakuliah yang dijadwalkan hanya satu kali (pada setiap pembuatan penjadwalan).
3. Penjadwalan dibuat untuk menjadwalkan matakuliah dengan pengulangan selama satu minggu.
4. Dalam satu waktu tidak boleh ada dosen yang mengajar di dua tempat.
5. Sebuah ruang tidak boleh dipakai oleh lebih dari atau matakuliah pada waktu yang bersamaan.
6. Matakuliah yang memesan waktu dan ruang, hanya bisa ditempatkan pada waktu dan ruang tersebut.
7. Matakuliah dengan bobot empat SKS dijadwalkan dengan dua kali pertemuan dalam seminggu, dan tidak ditempatkan pada hari yang bersamaan.
8. Matakuliah dengan bobot kurang dari empat SKS dijadwalkan dengan satu kali pertemuan dalam seminggu.
9. Hari aktif untuk perkuliahan adalah hari Senin hingga Jumat.



Gambar 2. Faktor-faktor yang Mempengaruhi Penjadwalan.



Gambar 3. Urutan Proses Sistem.

Sedangkan *soft constraint* yang diterapkan dalam penempatan jadwal matakuliah ini adalah:

1. Dosen dapat memesan waktu mengajar tertentu yang diinginkan.
2. Penempatan jadwal untuk waktu yang telah dipesan dosen disesuaikan dengan prioritas dosen.
3. Dosen dapat memilih untuk tidak mengajar pada waktu yang berurutan.
4. Pada waktu yang sama, tidak terdapat matakuliah yang mempunyai selisih kurang dari dua semester, karena diasumsikan untuk matakuliah yang mempunyai selisih semester kurang dari dua diambil oleh mahasiswa pada tingkat yang sama, sehingga akan ada kemungkinan terjadinya jadwal yang bentrok untuk mahasiswa.

Kriteria Jadwal Kuliah

Pada penelitian ini akan digunakan *constraint*, asumsi dan *rule* atau pengaturan penjadwalan kuliah sebagai berikut:

1. Ada sejumlah matakuliah yang ditawarkan pada satu semester yang ingin dibuat jadwalnya.
2. Setiap matakuliah yang dijadwalkan hanya satu kali (pada setiap pembuatan penajdwalan).
3. Penjadwalan dibuat untuk menjadwalkan matakuliah dengan pengulangan selama satu minggu.
4. Dalam satu waktu tidak boleh ada dosen yang mengajar di dua tempat.
5. Sebuah ruang tidak boleh dipakai oleh lebih dari atau matakuliah pada waktu yang bersamaan.
6. Matakuliah yang memesan waktu dan ruang, hanya bisa ditempatkan pada waktu dan ruang tersebut.
7. Jadwal hanya sampai hari jumat.
8. Jadwal berada dalam batas waktu dan ruang yang ditentukan.

Dari Diagram urutan proses pada Gambar 3, terdapat beberapa perancangan yatu perancangan data, perancangan proses dan perancangan antarmuka.

HASIL DAN PEMBAHASAN

Berdasarkan rancangan yang telah dibuat, diberikan beberapa skenario yang ditujukan untuk mengetahui fungsionalitas dari perangkat lunak yang dibuat. Terdapat lima skenario yang diujikan, yaitu:

1. Kemampuan algoritma koloni semut dalam penjadwalan dengan *Alfa*, *Beta* sesuai permintaan *user*.
2. Kemampuan AKS dalam penjadwalan dengan satu pemesanan jadwal.
3. Kemampuan AKS dengan beberapa pemesanan yang siap untuk dijadwalkan.
4. Kemampuan AKS dalam penjadwalan dengan beberapa pemesanan jadwal.
5. Kemampuan AKS dengan data ekstrim.
6. Kemampuan aplikasi untuk mendapatkan prosentase utilitas ruang dan beban sks dosen.

Uji Coba Skenario I

Dengan *setting* AKS berdasarkan permintaan *user* sebagai berikut:

1. Yang pertama dilakukan dengan *Alfa*: 1, *Beta*: 2 dan *Rho*: 0.5 maka menghasilkan sebagaimana terlihat pada Tabel 3.
2. Karena *alfa* dan *beta* sangat berpengaruh terhadap pembentukan produk *feromon*, tetapi produk *feromon* juga dipengaruhi oleh intensitas *feromon* maka secara tidak langsung nilai *Rho* juga mempengaruhi. Untuk itu sebelum dilakukan perhitungan produk *feromon* maka dilakukan perhitungan intensitas *feromon*, kemudian dilakukan perhitungan produk *feromon*: $a_{ij}(t) = [\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}$, dimana nilai η_{ij}

diperoleh dari posisi matakuliah yang menempati waktu *ij* dalam matriks *feromon*. Begitu pula pada data selanjutnya ukuran matriks yang terbentuk yaitu 6x6. Akumulasi Produk : Σ $a_{ij}(t)$, dimana $a_{ij}(t)$ merupakan nilai produk *feromon*.

3. Yang kedua dilakukan dengan *Alfa*: 2, *Beta*: 1 dan *Rho*: 0.5 maka menghasilkan sebagaimana pada tabel 4 produk *feromon*: $a_{ij}(t) = [\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}$, dimana η_{ij} diperoleh dari posisi matakuliah yang menempati waktu *ij* dalam matriks *feromon*. Begitu pula pada data selanjutnya ukuran matriks yang terbentuk yaitu 6x6. Akumulasi Produk : Σ $a_{ij}(t)$, dimana $a_{ij}(t)$ merupakan nilai produk *feromon* yang dihasilkan dari proses di atas. Sedangkan data dari perhitungan produk *feromon* di atas yang siap.

Dari percobaan ini terlihat bahwa komponen-komponen AKS yang berpengaruh dalam pembentukan nilai produk *feromon* adalah *alfa* dan *beta*. Untuk itu semakin besar nilai *alfa* dan *beta* maka semakin besar pula nilai produk *feromon* yang terbentuk pada setiap semut. Sedangkan nilai akumulasi produk bergantung pada nilai produk *feromon*. Semakin besar nilai produk *feromon* maka semakin besar pula nilai akumulasi produk. Sedangkan untuk mencapai jadwal yang optimal, nilai akumulasi produk sangat diperlukan dengan melakukan pencarian nilai akumulasi produk yang terkecil untuk dibandingkan dengan nilai akumulasi lain dan jika di antara keduanya terdapat nilai akumulasi produk terbesar, maka data tersebut ditunjukkan untuk melakukan proses selanjutnya.

Uji Coba Skenario II

Dengan permintaan memesan ruang sebagaimana Tabel 3. Data pemesanan jadwal tersebut merupakan salah satu semut dengan pemesanan *feromon* yang telah ditetapkan oleh *user*. Data semut lain yang siap untuk dijadwalkan dengan penguapan *feromon* yang kuat dan siap untuk dijadwalkan sesuai dengan Tabel 5.

Sedangkan data keluaran langsung dari proses AKS berupa bentuk tabel yang datanya masih berupa data numerik dapat ditunjukkan pada Tabel 4.

Hasil dari percobaan ini menunjukkan bahwa dengan melakukan pemesanan satu data jadwal maka data tersebut tersimpan dalam basisdata sebagai data yang siap dijadwalkan sesuai waktu dan ruang yang dipesan. Sedangkan semut lainnya yang tidak melalui proses pemesanan maka dilakukan proses perhitungan AKS untuk mendapatkan penguapan *feromon* yang kuat sehingga layak dijadwalkan.

Dengan adanya pemesanan jadwal maka data semut tersebut akan ditempatkan pada waktu dan ruang yang telah dipesan. Dan data semut lain yang tidak melalui pemesanan maka harus melalui proses perhitungan AKS untuk mendapatkan nilai *feromon* yang besar. Dimana nilai *feromon* didapatkan dari nilai akumulasi produk terbesar atau probabilitas terbesar yang mampu dijadwalkan.

Untuk mendapatkan penjadwalan yang optimal dengan AKS maka data-data yang akan diproses menjadi suatu jadwal harus memiliki nilai-nilai yang mempengaruhi penguapan *feromon* yang kuat.

Uji Coba Skenario III

Dengan beberapa pemesanan yang siap untuk dijadwalkan sebagai jadwal yang optimal yang ditunjukkan pada Tabel 5.

Setelah melakukan pemesanan seperti yang dilakukan pada Tabel 5 atau berdasarkan data yang ada maka merupakan semut dengan *feromon* yang telah ditetapkan berdasarkan pemesanan. Sedangkan untuk data semut-semut lain yang belum memiliki *feromon* maka dilakukan perhitungan dengan AKS ditunjukkan pada Tabel 6. Untuk data semut selanjutnya dapat ditunjukkan, sedangkan untuk perhitungan produk *feromon*: $a_{ij}(t) = [\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}$, dimana η_{ij} diperoleh dari posisi matakuliah yang menempati waktu *ij* dalam matriks *feromon*. Setelah produk *feromon* didapatkan maka langkah selanjutnya adalah perhitungan nilai akumulasi produk sebagai berikut: Akumulasi Produk: Σ $a_{ij}(t)$, dimana $a_{ij}(t)$ merupakan nilai produk *feromon* yang dihasilkan dari proses ini.

Dari hasil perhitungan pada Tabel 6 dapat diketahui bahwa untuk mendapatkan nilai penguapan *feromon* yang kuat, didapatkan dari nilai akumulasi produk yang terbesar atau probabilitas terbesar dan memenuhi sejumlah

constraint. Untuk lebih jelasnya ditunjukkan dalam Tabel 8.

Tabel 4. Solusi Hasil Jadwal dengan AKS.

Id_mk	Nip	Kelas	hari	jam	Ruang	Semester
CI1416	132296226	X	1	1	1	5
CI2413	051100001		1	1	3	1
CI1427	000011111	X	1	1	5	7
CI1407	132125657	A	1	1	6	3
CI2312	130532048		1	2	1	1
CI1402	132296226	C	1	2	2	3
CI1420	131996151	A	1	2	4	5
CI1426	132137825	X	1	2	6	7
CI1405	132306543	B	1	3	1	3
CI2313	131996151		1	3	2	1
CI1420	132318030	X	1	3	5	5
CI1427	132048148	A	1	3	6	7
CI1426	130532048	A	1	4	1	7
CI1405	051100016	C	1	4	2	3
CI2311	132230429		1	4	4	1
CI1419	132297166	B	1	4	5	5
CI2413	051100001		1	5	3	3
CI1408	132318030	C	1	5	3	3
CI1204	132306544		1	5	4	5
CI1425	131411100	X	1	5	5	7
CI1415	132303065	A	2	1	1	5
CI1524	131570363		2	1	2	7
CI1405	132048148	A	2	1	3	3
CI2311	132230429		2	1	4	1
CI1401	051100016	B	2	2	1	3
CI2313	131996151		2	2	2	1
CI1525	131570363		2	2	3	7
CI1418	132306296	A	2	2	6	5
CI2312	130532048		2	3	1	1
CI1417	051100012	X	2	3	3	5
CI1404	131411100		2	3	5	3
CI1425	051100001		2	3	6	7
CI1201	131996151	A	2	4	2	1
CI1403	131411100	A	2	4	4	3
CI1504	132125657		2	4	6	7
CI2101	132172210	B	2	5	1	1
CI1506	132306430		2	5	2	7
CI1417	051100016	A	2	5	4	5
CI1401	132048148	A	2	5	6	3
CI1403	132306544	C	3	1	3	3
CI1303	132172210	B	3	1	4	1
CI1303	132306544	C	3	2	5	1
CI1301	132048148	B	3	3	1	1
CI1301	132163671	C	3	4	2	1

Uji Coba Skenario IV

Setelah melakukan pemesanan seperti yang dilakukan pada Tabel 9 atau penetapan semut pada *feromon* sesuai keinginan *user* maka pemesanan jadwal tersebut sebagai proses pemesanan semut dengan penguapan *feromon* yang kuat. Sehingga pemesanan tersebut siap untuk dijadwalkan dengan pemenuhan

sejumlah *constraint*. Untuk semut lain yang belum mempunyai *feromon* yang kuat maka akan melalui proses AKS untuk mempunyai *feromon* yang kuat. Hal ini ditunjukkan dalam Tabel 10.

Dari hasil perhitungan, dapat diketahui bahwa untuk mendapatkan nilai penguapan *feromon* yang kuat, didapatkan dari nilai akumulasi produk yang terbesar atau probabilitas terbesar dan memenuhi sejumlah *constraint*. Untuk lebih jelasnya ditunjukkan dalam Tabel 8.

Tabel 9 menjelaskan bahwa data semut yang memiliki nilai akumulasi produk besar dan memenuhi sejumlah *constraint* maka semut tersebut merupakan solusi yang siap untuk dijadwalkan.

Dengan suatu peringatan tersebut maka pemesanan akan dapat dijadwalkan dengan pemesanan yang memiliki prioritas yang pertama pesan yang akan akan dapat dijadwalkan atau dapat juga untuk pemesanan pada prioritas kedua dilakukan perubahan waktu agar tidak terjadi bentrok.

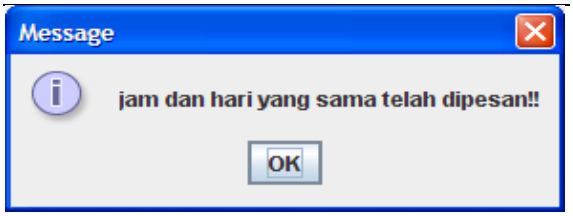
Uji Coba Skenario VI

Untuk mendapatkan total prosentase utilitas ruang maka perlu untuk menghitung prosentase penggunaan ruang tiap harinya. Tabel 10 menunjukkan proses perhitungan prosentase penggunaan ruang tiap hari.

Sedangkan untuk mendapatkan beban rata-rata SKS dosen maka perlu untuk mengetahui besarnya beban SKS untuk setiap dosen. Tabel beban SKS untuk setiap dosen ditunjukkan pada Tabel 11.

Tabel 3. Pemesanan Matakuliah.

Id_mk	NIP	Kelas	Hari	Jam	Ruang
CI1416	132296226	X	1	1	1



Gambar 4. *Allert* untuk Pemesanan Yang Bentrok Waktu.

Tabel 5. Pemesanan Beberapa Jadwal yang Optimal.

Id_mk	NIP	Kelas	Hari	Jam	Ruang
CI1303	130368610	A	4	2	2
CI1301	132048148	B	5	2	5
CI1418	132306296	A	4	1	1
CI1416	132296226	X	4	4	4
CI1415	132303065	A	5	1	1
CI1417	051100012	X	4	3	3

Tabel 6. Perhitungan Semut dalam Mendapatkan Penguapan *Feromon.*

Id_mk	NIP	Kelas	Hari	Jam	Ruang	Produk *Feromon*
CI2413	051100001		1	1	1	0.25
CI2311	132230429		1	1	2	1
CI2312	130532048		1	1	3	2.25
CI2313	131996151		1	1	4	4
CI1403	132306544	C	1	1	5	6.25
CI1403	131411100	A	1	1	6	9
CI2311	132230429		1	2	1	0.25
CI2312	130532048		1	2	2	1
CI2313	131996151		1	2	3	2.25
CI2413	051100001		1	2	4	4
CI1402	132296226	C	1	2	5	6.25
CI1427	000011111	X	1	2	6	9

Tabel 7. Pemesanan Beberapa Matakuliah dengan Menerapkan Sejumlah *Constraint* Dasar.

Id_mk	NIP	Kelas	Hari	Jam	Ruang
CI1504	132125657		1	2	3
CI1420	131996151	A	1	2	2
CI2313	131996151		2	1	1
CI2313	131996151		1	1	1
CI1416	132296226	X	5	1	1
CI1402	132296226	C	3	2	2
CI1598	132296226		4	1	1

Tabel 8. Perhitungan Produk *Feromon* dan Akumulasi Produk untuk Data yang Optimal.

Id_mk	NIP	Kelas	Hari	Jam	Ruang	Produk *Feromon*	Akumulasi Produk
CI1403	131411100	A	2	2	3	2.25	3.5

CI1417	051100016	A	2	1	4	4	7.5
CI1303	130368610	A	1	3	6	9	22.75
CI1420	132318030	X	1	3	2	1	1.25
CI1401	051100016	B	1	3	4	4	7.5
CI1408	132318030	C	1	5	5	6.25	13.75
CI1427	00001111	X	1	1	6	9	22.75
CI1525	131570363		2	2	4	4	7.5

Tabel 9. Pemesanan Matakuliah.

Id_mk	NIP	Kelas	Hari	Jam	Ruang
CI1420	131996151	A	1	1	2
CI2313	131996151		1	1	1

Tabel 10. Prosentase Penggunaan Ruang Tiap Hari.

Id_kelas	Hari	Prosentase (%)
1	1	60
	2	60
	3	20
	4	20
	5	20
2	1	60
	2	40
	3	40
	4	20
	5	0
3	1	20
	2	100
	3	0
	4	20
	5	0
4	1	80
	2	60
	3	0
	4	20
	5	0
5	1	100
	2	20
	3	0
	4	0
	5	20
6	1	60
	2	40
	3	0
	4	0
	5	0

Tabel 11. Beban SKS Dosen.

NIP	Nama	SKS
132306543	Imam Kuswardayan	3
131570363	Riyanarto Sarno	3
132163671	Dwi Sunaryonno	3
051100012	Mediana Aryuni	3
130368610	Supeno Djanali	3
132303065	Wahyu Suadi	3
132306430	Darlis Herumurti	3
132137825	Agus Zainal Arifin	3
132297166	Sarwosri	3
132306296	Tohari Ahmad	3
132172210	Yudhi Purwananto	5
132318030	Ahmad Saikhhu	5
132125657	Victor Hariadi	6
000011111	Daniel O Siahaan	6
132296226	Bilqis Amaliah	6
131411100	Much. Husni	8
051100001	Royyana Muslim I	8
132306544	Nunut Priyo Jatmiko	8
132048148	Suhadi Lili	9
051100016	Ahmad Khoirul Bashori	9
132230429	Fajar Baskoro	11
130532048	Handayani Tjandrasa	11
131996151	Joko Lianto Buliali	13

SIMPULAN

Berdasarkan analisis hasil ujicoba perangkat lunak maka dibuat simpulan sebagai berikut:

1. *Alfa, Beta* dalam AKS merupakan komponen-komponen yang berpengaruh dalam pembentukan nilai produk *feromon*. Sedangkan produk *feromon* diperlukan dalam pembentukan nilai akumulasi produk untuk mencapai jadwal yang optimal. Hal ini dibuktikan pada Uji Coba Skenario I.
2. AKS mampu menangani pemesanan satu data jadwal pada waktu tertentu dimana untuk *hard constraint* tidak boleh dilanggar sementara *soft constraint* boleh dilanggar (contoh *hard constraint* tidak boleh ada slot pada semester yang sama, satu waktu dengan dosen yang sama sedangkan contoh *soft constraint* dosen yang hanya ingin mengajar pada pukul 13.00) dengan dibuktikan pada Uji Coba Skenario II.
3. AKS mampu juga menangani beberapa pemesanan jadwal yang mampu untuk dijadwalkan sebagai solusi terakhir dari

penjadwalan dengan pemenuhan *constraint* yang ada. Ini dibuktikan dalam Uji Coba Skenario III.

4. Statistika yang dihasilkan oleh AKS menunjukkan bahwa AKS mampu menangani beberapa pemesanan jadwal dalam waktu tertentu dengan terpenuhinya sejumlah *constraint* sehingga membuat suatu aplikasi penjadwalan yang tidak bentrok. Ini dibuktikan pada Uji Coba Skenario IV.
5. AKS mampu menangani pemesanan jadwal dengan data ekstrim atau dengan mengabaikan *constraint* yang ada. Ini dibuktikan pda Uji Coba Skenario V.

DAFTAR PUSTAKA

[1] Maniezzo V. *Ant Colony Optimation*. Technical Report TR91-016. Milan: Politecnico di Milano. 2004.

[2] Arida L. *Penjadwalan Flow Shop Menggunakan Algoritma Koloni Semut*. Tugas Akhir. Surabaya: Teknik Informatika ITS. 2007.

[3] Darma K dan Widowati S. *Simulasi algoritma semut untuk penjadwalan ujian di STT Telkom*. Tugas Akhir. Bandung: STT Telkom. 2005.

[4] Dorigo M and Gambardella LM. Ant Algorithms for Discrete Optimization. Massachusetts Institute of Technology. *Artificial Life.* 5: 137-172. 1999.

[5] Dorigo M and Stützle T. *Ant Colony Optimization MIT Press.* 2004. URL: http://iridia.ulb.ac.be/mdorigo/ACO/ACO.html, diakses tanggal 6 September 2004.

[6] Nejsum MJ. *Methods for Interactive Constraint Satisfaction*. Openhagen: Department of Computer Science University of Openhagen. 2003.

[7] Wiriapradja G. *Aplikasi Penjadwalan Matakuliah dengan Menggunakan Algoritma Genetika dan Metode CSP pada Active Database, Studi Kasus di Jurusan Teknik Informatika ITS.* Tugas Akhir. Surabaya: Jurusan Teknik Informatika ITS. 2006.