IMPLEMENTASI ALGORITMA KOLONI SEMUT PADA PENJADWALAN MATAKULIAH UNIVERSITASBRAWIJAYA (Studi Kasus Jurusan Matematika Fakultas MIPA)

SKRIPSI

Oleh:
UNGGUL IZZA MUBAROK
0710960021-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2012

IMPLEMENTASI ALGORITMA KOLONI SEMUT PADA PENJADWALAN MATAKULIAH UNIVERSITASBRAWIJAYA (Studi Kasus Jurusan Matematika Fakultas MIPA)

Skripsi

Sebagai salah satu syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Oleh:
UNGGUL IZZA MUBAROK
0710960021-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2012

LEMBAR PENGESAHAN SKRIPSI

IMPLEMENTASI ALGORITMA KOLONI SEMUT PADA PENJADWALAN MATAKULIAH UNIVERSITASBRAWIJAYA (Studi Kasus Jurusan Matematika Fakultas MIPA)

oleh:
Unggul Izza Mubarok
0710960021-96

Setelah dipertahankan di depan Majelis Penguji
Pada tanggal 15Juni 2012
Dan dinyatakan memenuhi syarat untuk memperoleh gelar Sarjana Komputer dalam bidang Ilmu Komputer

Pembimbing I,	Pembimbing II,
Edy Santoso,S.Si.,M.Kom	**Dian Eka R,S.Si.,M.Kom**
NIP. 197404142003121004	**NIP. 19730619 2002122 001**

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Dr. Abdul Rouf Alghofari, M.Sc
NIP. 19670907 1992031 001

LEMBAR PERNYATAAN
Saya yang bertanda tangan di bawah ini :

Nama	**:** Unggul Izza Mubarok
NIM	**:** 0710960021-96
Jurusan	**:** Matematika
Program Studi	**:** Ilmu Komputer
Penulis skripsi berjudul	**:** Implementasi Algoritma Koloni Semut Pada Penjadwalan Mata Kuliah di Universitas Brawijaya (Studi Kasus Jurusan Matematika Fakultas MIPA)

Dengan ini menyatakan bahwa :

1. **Isi dari Skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam Skripsi ini.**
2. **Apabila dikemudian hari ternyata Skripsi yang saya tulis terbukti hasil jiplakan, maka saya akan bersedia menanggung segala resiko yang akan saya terima.**

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, 13 Juni 2012
Yang menyatakan,

Unggul Izza Mubarok
NIM. 0710960021-96

IMPLEMENTASI ALGORITMA KOLONI SEMUT PADA PENJADWALAN MATAKULIAH UNIVERSITASBRAWIJAYA (Studi Kasus Jurusan Matematika Fakultas MIPA)

ABSTRAK

Masalah penjadwalan kuliah merupakan masalah yang sangat kompleks dan rumit dimana inti dari masalah ini adalah bagaimana menjadwalkan berbagai komponen yang terdiri dari mahasiswa, dosen, dan waktu dengan memperhatikan sejumlah batasan dan syarat tertentu . Tujuan dari penjadwalan mata kuliah yaitu memimumkan jumlah pelanggaran *hardconstrain*. *Hardconstrain* tersebut meliputi dosen tidak boleh mengajar pada waktu yang bersamaan, mata kuliah dengan semester yang, sama prodi yang sama, kelas yang sama tidak boleh dijadwalkan di waktu yang bersamaan, pecahan SKS mata kuliah tidak boleh dijadwalkan pada hari yang sama, ruang perkuliahan tidak boleh digunakan untuk mata kuliah yang berbeda pada waktu yang bersamaan. Algoritma koloni semut merupakan salah satu algoritma heuristik yang dapat digunakan untuk menyelesaikan permasalahan penjadwalan mata kuliah. Berdasarkan hasil penelitian initidak terdapat pelanggaran untuk *hardcontrain*.. Nilai parameter algoritma yang terbaik adalah $\alpha = 3$ $\beta = 3$ $\rho = 0,5$ dan *pheronome* awal 0,01.

Kata Kunci : Algoritma koloni semut, Penjadwalan mata kuliah

ANT COLONY ALGORITHM IMPLEMENTATION ON BRAWIJAYA UNIVERSITY'S COURSE SCHEDULING (Case Studies In The Mathematic Majoring MIPA Faculty)

ABSTRACT

Course scheduling problem is a very complex and complicated problem which the point of this problem is how to schedule every componet such as students, lecturers, and time by observing some constraint. The purpose of course shceduling problem is minimazing number of clash in this case is minimazing the number of *hardconstrain*. Hardconstrain includes lecture should not teach at the same time, subject to the same semester, same study program, same class should not be scheduled at the same time, pieces of the course credit may not be scheduled on the same day, the clas should not be used for lectures differrent subjects at the same times.Algorithm ant colony is one of heuristic algorithm than can be used to solve course scheduling problem. According result of study there is no offense to hardconstrain. The best value of parameter algorithm is $\alpha = 3$ $\beta = 3$ $\rho = 0,5$ and the first pheronome 0,01.

Key word : Ant colony algorithm, course scheduling

KATA PENGANTAR

Puji syukur kehadirat Allah SWT, hanya dengan rahmat dan karunia yang telah diberikan kepada penulis, sehingga penulis dapat menyelesaikan skripsi yang berjudul "ImplementasiAlgoritma Koloni Semut Pada Penjadwalan Matakuliah Universitas Brawijaya (Studi Kasus Jurusan Matematika Fakultas MIPA)".

Skripsi ini disusun dan diajukan sebagai syarat untuk memperoleh gelar sarjana pada program studi Ilmu Komputer Universitas Brawijaya.

Dalam menyelesaikan skripsi ini, penulis telah mendapat bantuan dan dukungan dari banyak pihak. Untuk itu, penulis ingin menyampaikan penghargaan dan ucapan terima kasih kepada:

1. Edy Santoso, S.Si., M.Kom., dan Dian Eka, S.Si., M.Kom., selaku dosen pembimbing yang telah dengan bijaksana dan sabar dalam membimbing penyusunan skripsi ini.
2. Dr. Abdul Rouf Al-Ghofari, M.Sc., selaku Ketua Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya.
3. Drs. Marji, M.T., selaku Ketua Program Studi Ilmu Komputer, Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya.
4. Segenap Bapak dan Ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada penulis selama menempuh pendidikan di Program Studi Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya.
5. Secara khusus penulis ingin mengucapkan terima kasih kepada Ibunda yang penulis banggakan dan Ibunda tercinta serta kakak-adik penulis yang telah banyak memberikan dukungan dan pengorbanan baik secara moril maupun materiil sehingga penulis dapat menyelesaikan studi dengan baik.
6. Aldila Pratiwi, S. Ked yang selalu menemani, mengisi hari-hari dengan senyum dan memberikan semangat di saat suka maupun duka.

7. Segenab *Crew* cowmiscat 42A, kesumba 19, markas besar Sawojajar terima kasih atas dukungannya.
8. Semua sahabat di Program Studi Ilmu Komputer FMIPA Universitas Brawijaya yang telah banyak memberikan bantuan, dorongan serta motivasi sehingga skripsi ini dapat terselesaikan.
9. Semua pihak lain yang membantu untuk terselesaikannya skripsi ini yang tidak dapat disebutkan satu-persatu.

Semoga penulisan laporan skripsi ini bermanfaat bagi pembaca sekalian. Penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan sehingga penulis mengharapkan kritik dan saran yang membangun dari pembaca.

Malang, 13Juni 2012

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SOURCECODE

DAFTAR PERSAMAAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Penyampaian informasi pada lembaga pendidikan merupakan hal sangat penting, terutama informasi yang berkaitan dengan kegiatan perkuliahan. Perkuliahan merupakan suatu proses pertukaran informasi melalui pertemuan antara pengajar (dosen) dengan mahasiswa.Pertemuan ini biasanya dilakukan di suatu tempat dan waktu yang terbatas dan dibuat dengan aturan sistem kredit semester (SKS) yang waktunya dibatasi pula. Oleh sebab itu, agar perkuliahan berjalan tepat waktu maka dibutuhkan suatu penjadwalan mata kuliah yang mampu mengakomodasi keinginan mahasiswa dengan baik sehingga waktu yang terbatas tersebut bisa dimanfaatkan sebaik mungkin untuk menyelesaikan studi.

Pembuatan jadwal perkuliahan yang tepat merupakan salah satu faktor pendukung yang diperlukan.Diharapkan dengan adanya penjadwalan yang tepat, dapat membantu mahasiswa agar dapat mengambil mata kuliah sesuai dengan jumlah SKS yang harus ditempuh dan tidak mengalami kesulitan untuk pemilihan mata kuliah yang ingin ditempuh pada semester tersebut. Oleh sebab itu, perlu dibuat penjadwalan perkuliahan secara otomatis yang dapat membuat jadwal dengan cepat dan mudah sehingga proses pembuatan jadwal tersebut dapat diselesaikan dengan efisien.

Masalah penjadwalan kuliah merupakan masalah yang sangat kompleks dan rumit dimana inti dari masalah ini adalah bagaimana menjadwalkan berbagai komponen yang terdiri dari mahasiswa, dosen, ruang, dan waktu dengan memperhatikan sejumlah batasan dan syarat tertentu (*constraint*). Misalnya, dalam satu waktu tidak boleh ada dosen mengajar di dua tempat, begitu pula dengan mahasiswa. Dalam satu waktu mahasiswa tidak boleh melaksanakan perkuliah di dua tempat pula. Artinya, dalam pembuatan jadwal perkuliahan, harus diperhatikan faktor-faktor yang mempengaruhi pembentukan jadwal tersebut sehingga diperoleh solusi jadwal yang optimal.

Salah satu cara untuk melakukan optimasi penjadwalan perkuliahan yang kompleks dan rumit tersebut adalah dengan

menggunakan metode*heuristic*, yaitu metode yang memulai dari sebuah atau sekumpulan solusi awal, kemudian melakukan pencarian terhadap solusi yang lebih baik sehingga mendekati optimal. Kelebihan metode ini adalah tidak perlu menganalisa semua kemungkinan solusi untuk mendapatkan solusi yang optimal sehingga waktu penyelesaian yang dibutuhkan lebih cepat (Muttakhiroh, 2007). Salah satu algoritma yang digunakan adalah *ant colony optimization*(ACO) atau lebih dikenal dengan algoritma koloni semut. Algoritma ini mampu menghasilkan solusi yang optimal dan cepat untuk permasalahan yang sedemikian kompleks dan rumit seperti permasalahan penjadwalan kuliah. Artinya bahwa solusi optimal yang akan diperoleh adalah solusi yang mampu memenuhi keseluruhan *constraint* dari masalah penjadwalan tersebut dengan waktu komputasi yang relatif singkat.

Algoritma koloni semut diadopsi dari perilaku semut dalam mencari makanan. Secara alamiah semut mampu menemukan rute terpendek dalam perjalanan dari sarang menuju sumber makanan berdasarkan jejak kaki pada lintasan yang dilalui (Dorigo, 1996).Dalam aplikasinya, algoritma koloni semut merupakan teknik probabilitas untuk menyelesaikan masalah komputasi dengan menemukan jalur terpendek dari seluruh kemungkinan solusi yang ada.

Penelitian penjadwalan matakuliah dengan Algoritma koloni semut sebelumnya telah dilakukan oleh Antonio Fernandez di jurusan teknik elektro Universitas Diponegoro (2008), dimana metode pencarian kandidat solusi yang digunakan adalah*local search*. Algoritma ini merupakan algoritma yang belum lengkap karena pencarian solusi dapat dihentikan meskipun solusi yang ditemukan belum optimal. Oleh sebab itu, dalam penelitian ini akan digunakan metode *shortest path*, dimana sekumpulan solusi akan diubah dalam bentuk *graph*berbobot sedemikian sehinggadalam pencarian kandidat solusi yang paling optimal akan dihitung jarak terpendek dari kemungkinan yang ada.

Berdasarkan latar belakang yang telah diuraikan, maka dilakukan penelitian dengan judul "*Implementasi Algoritma Koloni Semut Pada Penjadwalan Mata Kuliah diUniversitas Brawijaya(Studi kasus Jurusan Matematika Fakultas MIPA Menggunakan)*".

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka permasalahan dapat dirumuskan menjadi :

1. Bagaimana mengimplementasikanalgoritma koloni semut pada penyusunan jadwal perkuliahan di Jurusan Matematika Fakultas MIPA Universitas Brawijaya.
2. Bagaimana pengaruh parameter α, β , ρ dan τ terhadap nilai pelanggaran dan waktu komputasi yang dibutuhkan oleh sistem untuk menghasilhan jadwal.

1.3 Tujuan

Tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut :

1. Mengimplementasikanalgoritma koloni semut pada penjadwalan perkuliahan di Jurusan Matematika Fakultas MIPA Universitas Brawijaya.
2. Mengetahui pengaruh parameter α, β ,ρ dan τ terhadap nilai pelanggaran dan waktu komputasi yang dibutuhkan oleh sistem untuk menghasilhan jadwal.

1.4 Batasan Masalah

Batasan permasalahan dari penelitianini adalah sebagai berikut :

1. Mata kuliah dengan bobot 4 SKS dipecah menjadi 2 pertemuan yaitu 2 SKS dan 2 SKS sedangkan untuk mata kuliah 3 SKS dipecah menjadi 2 pertemuan yaitu 2 SKS dan 1 SKS.
2. Dosen dan mahasiswa maksimal melakukan kegiatan perkuliahan sebanyak 4 kali dalam satu hari.
3. Tidak membedakan mata kuliah wajib maupun pilihan.

1.5 Manfaat

Manfaat yang dapat diperoleh dari penelitian ini yaitu:

- Dosen
  1. Mempermudah dosen dalam penyusunan jadwal kuliah.
  2. Menghindari kemungkinan dosen mengajar di dua tempat dalam satu waktu.

3. Menghindari kemungkinan dosen mengajar dengan jumlah SKS lebih dari standart yang di tentukan.
4. Memberi solusi dengan terbatasnya ruang yang bisa digunakan dalam perkuliahan

- Mahasiswa
  1. Memberi kemudahan mahasiswa dalam pemilihan mata kuliah yang akan diambil dalam satu semester.
  2. Menghindari kemungkinan mahasiswa gagal mengambil suatu mata kuliah karena bentrok dengan mata kuliah lain.

1.6 Sistematika Penulisan

Skripsi ini disusun dengan sistematika penulisan sebagai berikut:

BAB I : PENDAHULUAN
Membahastentang latar belakang, rumusan masalah, tujuan, manfaat, serta sistematika penulisan skripsi.

BAB II : TINJAUAN PUSTAKA
Berisi tentang teori yang menjadi acuan untuk pelaksanaan penelitian yang meliputi teori tentang penjadwalan dan algoritma semut dalam penyelesaian masalah.

BAB III : METODELOGI DAN PERANCANGAN
Membahas tentang segala sesuatu yang dibutuhkan untuk menyelesaikan masalah penjadwalan kuliah menggunakan algoritma semut.

BAB IV : HASIL DAN PEMBAHASAN
Berisi tentang implementasi aplikasi, pembahasan dan analisa hasil optimasi penjadwalan yang dihasilkan oleh algoritma semut.

BAB V : PENUTUP

Berisi kesimpulan dari sistem yang dirancang serta saran pengembangan dari keseluruhan tahapan pembuatan skripsi ini maupun pembuatan perangkat lunaknya.

BAB II
TINJAUAN PUSTAKA

2.1 Definisi Penjadwalan

Masalah penjadwalan dalam institusi pendidikan masih menjadi isu yang menarik dan secara luas masih diteliti oleh peneliti di dunia. Masalah penjadwalan diklasifikasikan dalam dua kategori utama, yaitu masalah penjadwalan kuliah dan masalah penjadwalan ujian. Masalah penjadwalan dalam kuliah merupakan masalah khusus dari masalah optimasi yang ditemukan dalam dunia nyata.

Penjadwalan adalah penempatansumber daya (*resource*) dalam satu waktu.Penjadwalan mata kuliah merupakanpersoalan penjadwalan yang umum dansulit, dimana tujuannya adalahmenjadwalkan pertemuan dari sumber daya yang ada, dimana sumber daya yang dimaksud adalah dosen pengasuh mata kuliah, mata kuliah,ruang kuliah, kelas mahasiswa, dan waktuperkuliahan (Pinedo, 1994).

Jadwal didefinisikan sebagai sesuatu yang menjelaskan dimana dan kapan orang-orang dan sumber daya berada pada suatu waktu (Chambers, 1992).Dalam kamus besar bahasa Indonesia Jadwal merupakan pembagian waktu berdasarkan rencana pengaturan urutan kerja.Jadwal juga didefiniskan sebagai daftar atau tabel kegiatan atau rencana kegiatan dengan pembagian waktu pelaksanaan yang terperinci. Sedangkan penjadwalan merupakan proses, cara, perbuatan menjadwalkan atau memasukan dalam jadwal.

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

2.1.1 Penjadwalan Kuliah Standart Internasional

Masalah penjadwalan kuliah standar internasional dikenal dengan *University Course Timetabling Problem* (UCTP). Penjadwalan ini akan menjadwalkan sejumlah kuliah dengan 45 slot

waktu yakni 9 slot waktu perhari dari senin sampai jumat. Komponen yang mempengaruhi terdiri dari kapasitas ruangan yang tersedia, waktu kehadiran, dosen, mahasiswa, pengikut kuliah serta batasan-batasan lain yang sudah ditentukan.

Batasan atau kendala yang terdapat dalam UCPT terbagi dalam dua kategori yaitu kendala *hard* dan kendala *soft*.Kendala *hard* merupakan kendala esensial yang harus dipenuhi dalam pembuatan jadwal, sedangkan kendala *soft* merupakan kendala non esensial yang menentukan kualitas dari jadwal yang dihasilkan. Masalah kendala ini akan dibahas lebih khusus dalam landasan teori ini.

Kendala *hard* dan kendala *soft* terjadi akibat interaksi antar komponen jadwal, seperti kapasitas ruang dan jumlah mahasiswa pengikut kuliah. Selain itu, interaksi juga dapat terjadi antar kuliah-kuliah yang tidak boleh dijadwalkan pada waktu yang bersamaan.Klasifikasi kedua kendala tersebut telah ditentukan oleh Universitas Napier karena disesuaikan dengan kondisi penjadwalan kuliah di universitas tersebut (Rossidoria, 2004).

Dalam UCPT, untuk membentuk solusi jadwal kuliah yang layak harus dipenuhi beberapa aturan sebagai berikut :

1. H_1 : Dua atau lebih kuliah tidak dapat diberikan pada suatu waktu dengan ruangan yang sama.
2. H_2 : Setiap kuliah harus dilaksanakan pada ruangan yang memenuhi fasilitas dan kapasitas untuk kuliah tersebut.
3. H_3 : Tidak ada mahasiswa yang mendapat 2 atau lebih kuliah pada waktu yang sama.

Dimana H_i (i = 1, 2, 3) adalah kendala *hard*.

Setelah seluruh kendala *hard*terpenuhi dalam pembentukan jadwal, maka akan dilanjutkan untuk meminimumkan fungsi penalti kendala *soft*yakni pelanggaran jadwal terhadap kendala *soft*. Sedapat mungkin kendala *soft* harusdiminimalisasi karena sangat menentukan kualitas dari jadwal tersebut.

1. S_1 : Mahasiswa tidak memperoleh kuliah pada akhir slot waktu terakhir setiap hari.
2. S_2 : Mahasiswa tidak mendapat kuliah lebih dari 6 jam berturut-turut.
3. S_3 : Mahasiswa tidak memperoleh hanya satu kuliah dalam 1 hari.

Dimana S_i (i = 1, 2, 3) adalah kendala *soft*.

2.1.2 Mesin Penjadwalan Kuliah

Dari permasalahan penjadwalan kuliah yang rumit, dirancang suatu mesin penjadwalan dengan tujuan membantu mempermudah penyusunan jadwal tersebut.Kemudahan yang dimaksud adalah kecepatan dan keakuratan jadwal yang diperoleh jika dibandingkan dengan penjadwalan yang dibuat secara manual.

Terdapat beberapa metode pendekatan yang digunakan untuk pembuatan jadwal yang optimal, dimana metode-metode tersebut harus mengacu pada aturan penalti dari kendala *soft* dihitung dengan persamaan 2.1.

$$P^j = \sum_{i \in I} \frac{f_i(J_i^j) - f_i(J_i^b)}{f_i(J_i^b) - f_i(J_i^b)} \qquad (2.1)$$

dimana : i: jumlah instansi
j: Partisipan ke – j
f_i: Penalti *soft*instansi ke – i
J_i^j : Jadwal yang disusun oleh partisipan *j* untuk instansi *i*
J_i^b : Jadwal terbaik pada instansi ke-*i* yang telah diketahui
J_i^w : Jadwal terburuk pada instansi ke-*i* yang telah diketahui

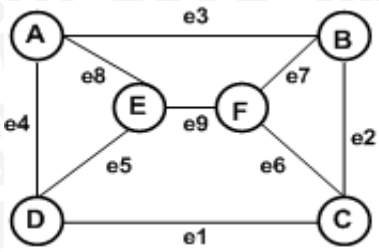
Dari persamaan 2.1, penjadwalan dikatakan optimal jika memiliki nilai penalti minimum. Metode yang menghasilkan nilai penalti minimum tersebut adalah metode *heuristic*(Rossidoria, 2004).

2.2 Teori Graph

2.2.1 Definisi Graph

Definisi 2.1 (**Wilson, R.J dan Watkhins, J.J, 1990**)

Suatu Graph G terdiri atas himpunan tidak kosong dari elemen-elemen yang disebut titik (vertek), dan suatu daftar pasangan vertek yang tidak terurut disebut sisi (*edge*). Himpunan vertek dari suatu graph G dinotasikan V dan daftar himpunan edge dari graph G dinotasikan sebagai E. untuk selanjutnya Graph G dinotasikan dengan G = (V, E).



Gambar 2.1 Contoh graph G

Gambar 2.1 menunjukkan graph G dengan V = {A, B, C, D, E, F}danE = {e1, e2, e3, e4, e5, e5, e6, e7, e8, e9}.

Definisi lain dari Graph adalah kumpulan simpul (*nodes*) yang dihubungkan satu sama lain melalui sisi atau busur (*edges*). Suatu graph G terdiri dari himpunan V dan himpunan E (Muntakhiroh, 2007).

a. Verteks (simpul) : V = himpunan simpul yang terbatas dan tidak kosong.
b. Edge (sisi/busur) : E = himpunan busur yang menghubungkan sepasang simpul.

Simpul-simpul pada graph dapat merupakan obyek sembarang seperti kota, atom pada zat kimia, mata kuliah dan sebagainya. Busur dapat menunjukkan hubungan (relasi) sembarang seperti rute penerbangan, jalan raya, sambungan telpon dan lain-lain. Notasi graf:G(V, E) artinya graf G memiliki V simpul dan E busur.

2.2.2 Definisi Walk

Definisi 2.2 **(Evans, J.R dan Edward, M, 1992)**

Suatu walk (jalan) dalam graph adalah barisan vertek-vertek dan edge-edge yang dimulai dan diakhiri oleh suatu vertek. Panjang suatu walk dihitung berdasarkan jumlah edge dalam walk tersebut.

Walk juga dapat diartikan sebagai suatu perjalanan (dalam sebuah graph) dari vertek satu ke vertek lain yang terhubung dengan suatu edge.

2.2.3 Definisi Trail dan Path

Definisi 2.3 **(Wilson, R.J dan Watkhins, J.J, 1990)**

Walk yang panjangnya k pada suatu graph G adalah urutan k edge G yang berbentuk

uv, vw, wx, ….., yz.

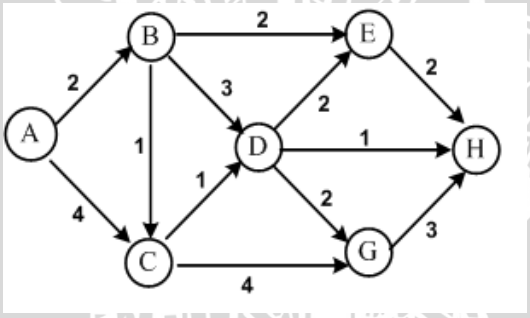
Walk ini dinotasikan dengan uvwx…..yz, dan ditunjuk sebagai walk antara u dan z. jika semua edge (tetapi tidak perlu semua vertek) suatu walk berbeda, maka walk itu disebut **trail**. Jika semua vertek pada trail berbeda, maka trail itu disebut dengan **path**.

2.2.4 Macam-macam Graph Menurut Arah dan Bobotnya

Menurut arah dan bobotnya, graph dibagi menjadi empat bagian, yaitu : graph berarah dan berbobot, graph tidak berarah dan berbobot, graph berarah dan tidak berbobot serta graph tidak berarah dan berbobot (Leksono, 2009).

2.2.4.1 GraphBerarah dan Berbobot

Graph berarah dan berbobot jika tiap busur mempunyai anak panah dan bobot. Gambar 2.2 menunjukkan graph berarah dan berbobot yang terdiri dari tujuh titik yaitu : titik A, B, C, D, E, G, H. Titik menunjukkan arah ke titik B dan titik C, titik B menunjukkan arah ke titik D dan titik C dan seterusnya. Bobot antara titik A dan B pun sudah diketahui.



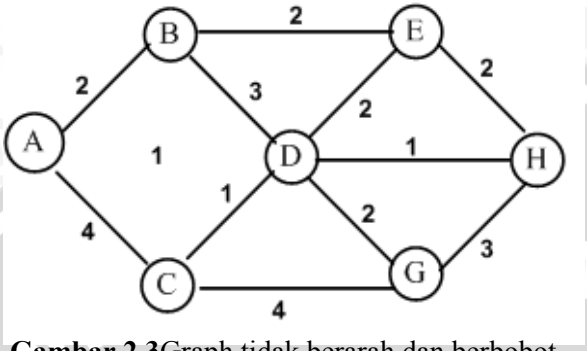
Gambar 2.2 Graph berarah dan berbobot

Vertek A mempunyai dua edge yang masing-masing berhubungan dengan vertek B dan C dimana masing-masing vertek

memiliki bobot tertentu, vertek B memiliki 3 edge yaitu vertek C, vertek D, vertek E dan masing-masing mempunyai bobot tertentu.

2.2.4.2 Graph Tidak Berarah dan Berbobot

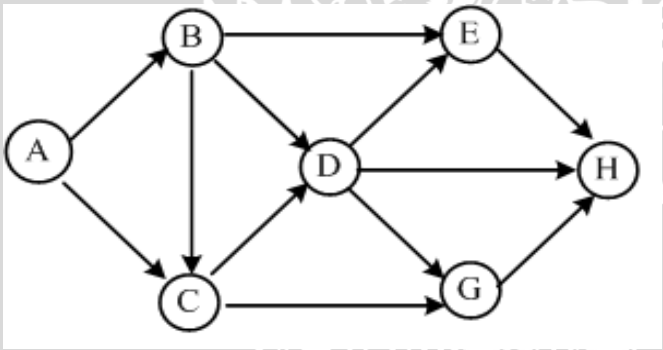
Graph Tidak berarah dan berbobot jika tiap busur tidak mempunyai anak panah, tetapi mempunyai bobot. Gambar 2.3 menunjukkan graf tidak berarah dan tidak berbobot. Graph terdiri dari tujuh titik yaitu A, B, C, D, E, G, H. Titik A tidak menunnjukan arah ke titik B dan C, namun bobot antara titik A dan B telah dikatahui. Begitu pula dengan titik yang lain.



Gambar 2.3 Graph tidak berarah dan berbobot

2.2.4.3 GraphBerarah dan Tidak Berbobot

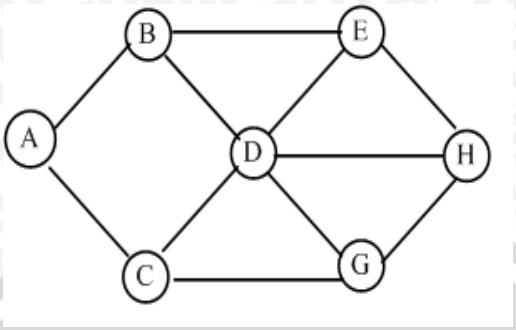
Graph Bearah dan tidak berbobot jika tiap busur atau edge mempunyai anak panah yang tidak berbobot. Gambar 2.4 menunjukkan graph berarah dan tidak berbobot.



Gambar 2.4 Graph berarah dan tidak berbobot

2.2.4.4 Graph Tidak Berarah dan Tidak Berbobot

Graph tidak berarah dan tidak berbobot jika tiap busur atau tiap edge tidak mempunyai panah dan tidak berbobot. Gambar 2.5 menunjukkan graph yang tidak berarah dan berbobot.



Gambar 2.5 Graph tidak berarah dan tidak berbobot

2.2.5 Matriks kedekatan (*Adjacency Matrix*)

Suatu graph dengan simpul sebanyak *n*, maka matriks kedekatan mempunyai ukuran $n \times n$ (*n* baris dan *n* kolom). Jika antara dua buah simpul terhubung maka elemen matriks bernilai 1 dan sebaliknya bernilai 0 jika tidak terhubung. Tabel matriks kedekatan untuk graph ABCDEGH pada subbab 2.2.4.3 dapat dilihat pada tabel 2.1 berikut.

Tabel 2.1 Matriks kedekatan graph G

	A	**B**	**C**	**D**	**E**	**G**	**H**
A	**0**	1	1	**0**	**0**	**0**	**0**
B	1	**0**	**0**	1	1	**0**	**0**
C	1	**0**	**0**	1	**0**	1	**0**
D	**0**	1	1	**0**	1	1	1
E	**0**	1	**0**	1	**0**	**0**	1
G	**0**	**0**	1	1	**0**	**0**	1
H	**0**	**0**	**0**	1	1	1	**0**

2.3 Optimasi

Penjadwalanmata kuliah merupakan masalah yang sangat rumit untuk diselesaikan.Secara tidak langsung menyelesaikan masalah penjadwalan sangat erat kaitannya dengan masalah optimasi yaitu kondisi dimana setiap solusi yang didapat belum bisa dibuktikan kebenarannya bahwa solusi tersebut adalah optimal. Namun, solusi tersebut akan menghasilkan solusi yang baik dengan memotong permasalahan yang kompleks menjadi lebih sederhana dengan tujuan meminimalkan waktu penyelesaian sehingga solusi tersebut masih bisa dioptimalkan lagi.

2.3.1 Definisi Masalah Optimasi

Optimasiadalah suatu proses untuk mencapai hasil yang optimal (nilai efektif yang dapat dicapai). Dalam disiplin matematika optimasi merujuk pada studi permasalahan yang mencoba untuk mencari nilai minimal atau nilai maksimal dari suatu fungsi riil. Untuk dapat mencapai nilai optimal baik minimal maupun maksimal tersebut, secara sistematis dilakukan pemilihan nilai variabel *integer* atau riilyang akan memberikan solusi optimal (Wardy, 2007).

Optimasi merupakan masalah dengan memaksimalkan atau meminimalkan suatu besaran tertentu, yang disebut dengan fungsi objektif.Fungsi objektif ini bergantung pada sejumlah variabel.Variabel-variabel ini tidak saling berhubungan atau saling bergantung melalui satu atau lebih kendala (Bronson, 1983).

Optimasi terbagi menjadi dua yaitu optimasi linear dan nonlinier.Permasalahan optimasi dikatakan nonlinier jika fungsi objektif dan kendalanya mempunyai bentuk nonlinier pada salah satu atau keduanya (Luknanto, 2000).

2.3.2 *Definisi* Fungsi Objektif

Fungsi objektif $f : X \rightarrow Y$ dengan $X \epsilon L$ dan $Y \epsilon R$ adalah fungsi untuk masalah yang dioptimalkan. Dengan L merupakan himpunan lintasan yang mungkin terjadi (Weise, 2008).

2.3.3 Definisi Nilai Optimal

Nilai optimal adalah nilai yang didapat melalui suatu proses dan dianggap menjadi solusi jawaban yang paling baik dari semua solusi yang ada (Wardy, 2007).

2.3.4 Macam-macam Permasalahan Optimasi

Permasalahan yang berkaitan dengan optimasi sangatlah kompleks dalam kehidupan sehari-hari.Nilai optimal yang didapat dalam optimasi dapat berupa besaran panjang, waktu, jarak, dan lain-lain. Berikut ini adalah beberapa persoalan optimasi:

1. Menentukan lintasan terpendek dari suatu tempat ke tempat lain.
2. Menentukan jumlah pekerja seminimal mungkin dalam melakukan proses produksi agar pengeluaran biaya pekerja dapat diminimalkan dan hasil produksi tetap maksimal.
3. Mengatur rute kendaraan umum agar semua lokasi dapat dijangkau.
4. Mengatur *routing*jaringan kabel telepon agar biaya pemasangan kabel tidak terlalu besar dan penggunaanya tidak boros.
5. Mengatur penjadwalan mata kuliah agar tidak terjadi bentrok antara mata kuliah, dosen pengajar sehingga mahasiswa bisa mengambil mata kuliah yang ditawarkan tersebut.

Selain permasalahan di atas, masih banyak permasalahan lain yang terdapat diberbagai bidang.

2.3.5 Penyelesaian Masalah Optimasi

Dalam masalah optimasi, yaitu penjadwalan mata kuliah, merupakan masalah optimasi yang berkaitan dengan pencarian jalur terpendek atau jalur tercepat dalam menemukan solusi (*foranging*). Masalah optimasi ini dapat diselesaikan dengan dua metode, yaitu Metode konvensional dan metode heusristik (Muttakhiroh, 2007).

2.3.5.1 Metode Konvensional

Metode konvensional adalah Metode yang menggunakan perhitungan matematika biasa.Metode konvensional yang biasa digunakan adalah algoritma Djikstra, algoritma Floyd-Warshall atau algoritma Bellman-Ford.

2.3.5.2 Metode Heuristik

Metode heuristik adalah metode yang memulai dari sebuah atau sekumpulan solusi awal, kemudian melakukan pencarian terhadap solusi yang lebih baik sehingga mendekati optimal.Metode ini sangat baik digunakan untuk menyelesaikan masalah optimasi kombinatorial yang rumit, dimana metode konvensional atau perhitungan manual matematika sudah tidak mampu lagi untuk menyelesaikannya atau dengan kata lain permasalahan tersebutsangat sulit untuk diformulasikan (Muttakhiroh, 2007).

Kelebihan suatu algoritma dengan metode heuristik adalah tidak perlu menganalisa semua kemungkinan solusi untuk mendapatkan solusi yang optimal (seperti algoritma dengan metode konvensional), sehingga solusi yang mendekati solusi optimal dapat diperoleh dalam waktu komputasi yang singkat. Dengan kata lainmetode ini adalah teknik analisa yang digunakan untuk meningkatkan kinerja melalui proses komputasi yang dirancang untuk menyelesaikan masalah tanpa pembuktian benar tidaknya solusi yang diberikan. Namun solusi yang dihasilkan biasanya merupakan solusi yang akurat dan optimal (Muttakhiroh, 2007).

Beberapa metode heuristik yang biasa digunakan adalah algoritma genetika, logika fuzzy (*fuzzy logic*),jaringan syaraf tiruan (JST),*tabu search*, *simulated annealing* dan algoritma koloni semut (*ant colony optimization*).

Dalam skripsi ini, yaitu masalah penjadwalan mata kuliah di jurusan matematika Fakultas MIPA Universitas Brawijaya Malang, digunakan metode haeuristik dengan algoritma yang digunakan adalah algortima koloni semut.

2.4 Algoritma Koloni Semut

2.4.1 Sejarah Algoritma Koloni Semut

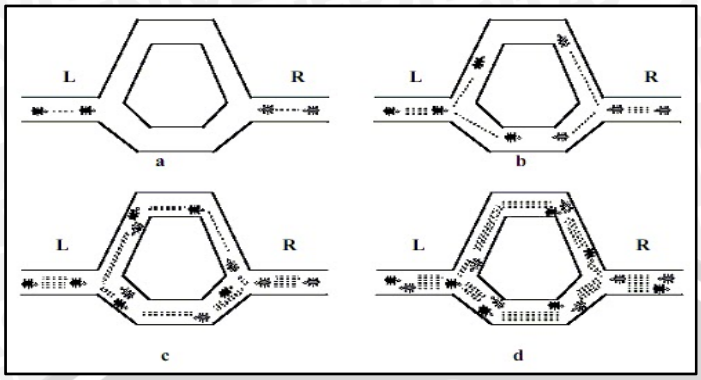
Algoritma semut diperkenalkan oleh **Moyson** dan **Manderick** dan secara luas dikembangkan oleh **Marco Dorigo,** merupakan teknik probabilistik untuk menyelesaikan masalah komputasi dengan menemukan jalur terbaik. Algoritma ini diambil dengan analogi perilaku semut dengan menemukan jalur dari koloninya menuju sumber makanan.

2.4.2 Cara Kerja Algoritma Koloni Semut Mencari Jalur Optimasi

Algoritma koloni diadopsi dari perilaku semut yang dikenal sebagai sistem semut yaitu sebuah probabilistik komputasi teknik untuk memecahkan masalah yang dapat digunakan untuk menemukan jalur terbaikmelalui grafik (Dorigo, 1996). Semut mampu mengindera lingkungannya yang kompleks untuk mencari makanan dan kemudian kembali ke sarangnya dengan meninggalkan zat *pheromone*pada jalur-jalur yang mereka lalui. *Pheromone*adalah zat kimia yang berasal dari kelenjar endokrin dan digunakan oleh mahluk hidup untuk mengenali sesama jenis, individu lain, kelompok dan untuk membantu proses reproduksi. Berbeda dengan hormon, *pheromone*menyebar ke luar tubuh dapat mempengaruhi dan dikenali oleh individu lain yang sejenis (satu spesies). Proses peninggalan *pheromone* ini dikenal dengan *stigmergy*, sebuah proses memodifikasi lingkungan yang tidak hanya bertujuan untuk mengingat jalan pulang ke sarang, tetapi juga memungkinkan para semut berkomunikasi dengan koloninya (Fernandez, 2008).

Secara alamiah koloni semut mampu menemukan rute terpendek dalam perjalanan dari sarang ke tempat-tempat sumber makanan berdasarkan jejak *pheromone* yang disekresikan lewat tubuhnya pada lintasan yang telah dilalui. Semakin banyak semut yang melalui suatu lintasan, maka akan semakin jelas bekas jejak kakinya. Hal ini akan menyebabkan lintasan yang dilalui semut dalam jumlah sedikit, semakin lama semakin berkurang kepadatan semut yang melewatinya, atau bahkan tidak dilewati sama sekali. Sebaliknya lintasan yang dilalui semut dengan jumlah banyak, semakin lama akan semakin bertambah kepadatan semut yang melewatinya, bahkan semua semut akan melalui lintasan tersebut (Fernandez, 2008).

Gambar 2.6 menjelaskan perjalanan semut dalam menemukan jalur terpendek dari sarang ke sumber makanan. Terdapat beberapa proses sehingga sumut sampai pada sumber makanan dan kembali lagi ke sarang.



Gambar 2.6 Perjalanan semut menemukan sumber makanan

1. Gambar 2.6 (a) menjelaskan ada dua kelompok semut yang akanmelakukan perjalanan. Kelompok L yaitu kelompok yang berangkatdari arah kiri yang merupakan sarang semut dan kelompok R yang berangkat dari kanan yang merupakan sumber makanan.
2. Gambar 2.6 (b) menjelaskan kedua kelompok semut dari titik berangkat sedang dalam posisi pengambilan keputusanjalan sebelah mana yang akan diambil. Kelompok semut L membagi duakelompok lagi. Sebagian melaluijalan atas dan sebagian melalui jalan bawah. Demikian pula dengan kelompok semut R.
3. Gambar 2.6 (c) menjelaskan kelompok semut berjalan pada kecepatan yang sama denganmeninggalkan *pheromone*di jalan yang telah dilalui. *Pheromone*yangditinggalkan oleh kumpulan semut yang melalui jalan atas telah mengalamibanyak penguapan atau evaporasi*pheromone*karena semut yang melalui jalan atas berjumlah lebih sedikitdari pada jalan yang di bawah. Hal ini dikarenakan jarak yang ditempuh lebih panjang daripada jalan bawah. Sedangkan *Pheromone* yang berada di jalan bawah,

penguapannya cenderung lebih lama karena semut yang melalui jalan bawah lebih banyak daripada semut yang melalui jalan atas.

4. Gambar 2.6 (d) menjelaskan semut-semut yang lain pada akhirnya memutuskan untuk melewati jalan bawah karena *pheromone* yang ditinggalkan masih banyak. Sedangkan *pheromone* pada jalan atas sudah banyak menguap sehingga semut-semut tidak memilih jalan atas tersebut. Semakin banyak semut yang melalui jalan bawah maka semakin banyak semut yang mengikutinya. Demikian juga dengan jalan atas, semakin sedikit semut yang melalui jalan atas, maka *pheromone*yang ditinggalkan semakin berkurang bahkan hilang. Dari sinilah kemudian terpilihlah jalur terpendek antara sarang dan sumbermakanan.

2.4.3 Metode Algoritma Koloni Semut

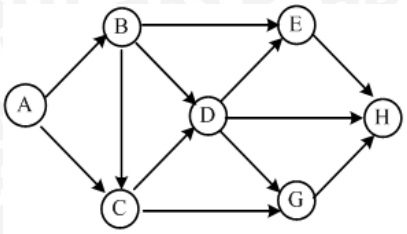
Algoritma semut digunakan untuk memperoleh solusi yang optimal dari sebuah permasalahan yang rumit. Algoritma ini berjalan secara kontinu dan beradaptasi dengan cepat untuk menangani perubahan yang *realtime*sehingga efisiensi pencarian akan memperoleh solusi yang optimal.

Ada beberapa metode dalam algoritma koloni semut yang bisa digunakan sehingga memperoleh hasil pencarian yang optimal, yaitu:

2.4.3.1 Foranging

Metode *foranging* adalah salah satu metode dari algoritma koloni semut yang ditekankan pada pencarian jalur tercepat dari permasalahan (*shortest path problem*) dalam menentukan solusi.Metode ini menerapkan tingkah laku semut ketika mencari makanan (Leksono, 2009).

Jalur tercepat (*shortest path*) adalah suatu jaringan pengarah perjalanan dimana seorang pengarah jalan ingin menentukan jalur terpendek antara dua kota berdasarkan beberapa jalur alternatif yang tersedia, dimana titik tujuannya hanya satu. Gambar 2.7 adalah contoh graph berarah dan tidak berbobot.



Gambar 2.7 Graf ABCDEGH

Pada gambar 2.7, misalkan dari kota A ingin menuju kota G. Untuk menuju kota G, dapat dipilih beberapa jalur yang tersedia :

A – B – C – D – E – H – G
A – B – C – D – E – G
A – B – C – D – G
A – B – C – G
A – B – D – E – H - G
A – B – D – G
A – B – E – D–G
A – C – D – E – H – G
A – C – D – G
A – C –G

Berdasarkan data diatas, dapat dihitung jalur terpendek dengan mencari jarak antara jalur-jalur tersebut. Apabila jarak antara jalur belum diketahui, jarak antar kota tersebut dapat dihitung berdasarkan koordinat kota-kota tersebut, kemudian menghitung jarak terpendek yang dapat dilalui.

Dalam algoritma semut, Secara umum semua semut akan menyebar secara acak menuju setiap kota yang ada sehingga sampai ke kota tujuan G dan kota B, C, D, E, merupakan kota-kota perantara sehingga setiap semut mampu menuju ke G. Selanjutnya jika masing-masing semut yang secara acak sudah sampai kota G, maka semut dengan waktu tercepat samapai ke kota A akan diikuti oleh semut-semut lain sehingga jejak *pheromone* yang melekat akan semakin kuat. Metode ini merupakan bagian dari algoritma koloni semut dengan teknik *foranging* atau jarak terpendek (Leksono, 2009).

2.4.3.2 Division of labor

Metode *Division of labor*mengimplementasikan tingkah laku semut dalam membagi pekerjaan ke dalam bagian-bagian tertentu dan dengan hak-hak tertentu. Seperti yang kita tahu terdapat berbagai jenis semut, mulai dari semut pengumpul makanan, semut petarung, semut peranakan.Semuanya memiliki tugas masing-masing dan terpisah, hal inilah yang diterapkan dalam metode ini untuk menempatkan tugas tertentu dengan metode algoritma tertentu agar lebih optimal. Biasanya metode ini digunakan dalam masalah penugasan (Zukhri, 2005).

Masalah penugasan (MP) merupakan permasalahan yang membutuhkan beban dan waktu komputasi yang cukup berat.Secara umum, masalah penugasan berkaitan dengan keinginan perusahaan dalam mendapatkan pembagian atau alokasi tugas yang optimal, dalam arti apabila penugasan itu berkaitan dengan keuntungan, maka bagaimana alokasi tugas atau penugasan tersebut memberi keuntungan yang optimal, begitu pula sebaliknya jika itu menyangkut masalah biaya (Zukhri, 2005).

Dalam sudut pandang optimasi, pemecahan masalah penugasan bertujuan untuk menyusun pasangan sumberdaya dan aktifitas berdasarkan penugasan satu ke satu, sedemikian sehingga dapat menghemat total biaya yang dibutuhkan pada kasus minimasi atau mendapatkan keuntungan yang paling besar pada kasus maksimasi, sehingga secara manual untuk menemukan solusi harus diperiksa kombinasi sebanyak faktorial dari banyak sumber daya atau faktorial dari banyak aktifitas. Masalah penugasan harus dilakukan penjadwalan terhadap *n* sumberdaya untuk menangani *n* aktifitas, sedemikian sehingga biaya yang dibutuhkan untuk semua aktifitas adalah minimum(Zukhri, 2005).

Algoritma koloni semut dalam masalah penugasan ini, dengan melakukan modifikasi pada beberapa konsep karena masalah penugasan berbeda dengan masalah TSP. Solusi yang dicari dalam masalah penugasan adalah pasangan sumber daya dan aktifitas. Dalam solusinya, perjalanan semut harus disesuaikan, harus dimodelkan sedemikian sehingga menjadi masalah pencarian rute. Hal ini bisa dilakukan dengan menganggap koloni semut harus melakukan perjalanan dari sebuah kota menuju kota lainnya

menggunakan biro jasa transportasi. Setiap semut tidak boleh menggunakan biro jasa yang samauntuk kota yang berbeda. Banyak kota dan banyak biro jasa transortasi ditentukan oleh banyak aktifitas dan banyak sumber daya dalam matriks biaya. Banyak kelompok kota ditentukan berdasarkan mana yang lebih kecil antara banyak aktifitas dan sumber daya. Yang lebih kecil akan menjadi banyak kelompok kota, sedangkan yang lebih besar akan menentukan banyak biro jasa (Zukhri, 2005).

Konsep jarak antar kota juga harus disesuaikan dalam metode ini. Pengertian jarak disini dapat dianalogikan dengan tarif biro jasa transportasi yang harus 'dibayar' oleh semut. Ongkos ini tidak tergantung pada jarak riil ke kota tersebut, tetapi justru tergantung pada kota yang akan dituju, dari manapun berangkatnya. Adanya keterlibatan biro jasa dalam pendekatan algoritma koloni semut metode ini juga dengan sendirinya merubah konsep jejak antar kota.

2.4.3.3 Cemetery organization and brood sorting

Metode *Cemetery organization and brood sorting* menerapkan tingkah laku semut dalam hal kerja sama antar semut. Ketika semut membangun sarang atau ketika akan bertelur, mereka akan menyusun secara rapi dan teratur serta dilakukan secara bersama agar cepat selesai. Hal ini diterapkan dalam algoritma untuk pengolahan *parallel* maupun *clustering*, dengan menggunakan tenaga banyak untuk mengerjakan sebuah *task* maka pekerjaan akan cepat selesai (Fernandez, 2008).

2.4.3.4 Cooperative transport

Metode*Cooperative transport*, metode ini menggabungkan banyak kemampuan dari semut untuk mendapatkan suatu sistem. Metode ini sering digunakan dalam pembuatan robot, dimana setiap gerakan robot harus diatur sedemikian rupa sehingga kerja sama antara satu bagian dengan bagian lainnya akan tetap sinkron dan sesuai keinginan (Fernandez, 2008).

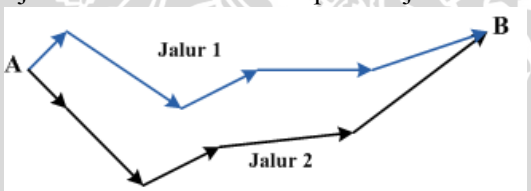
2.4.4 AnalisisaAlgoritma Koloni Semut Untuk Mencari Nilai Optimal Menggunakan Graph

Untuk mendiskusikan algoritma koloni semut, lingkungan yang akan digunakan adalah sebuah graph yang *fully connected* (setiap node memiliki baris ke node yang lain) dan*bidirectional*(setiap jalur bisa ditempuh bolak balik dua arah). Setiap garis memiliki bobot yang menunjukkan jarak antara dua titik yang dihubungkan oleh garis tersebut (mindaputra, 2009).

Algoritma ini menggunakan sistem multi agen, yang berarti akan dikerahkan seluruh koloni semut yang masing-masing bergerak sebagai agen tunggal. Setiap semut menyimpan *tabu list*yang memuat titik yang pernah dilalui, dimana semut tidak diijinkan untuk melalui titik yang sama dua kali dalam satu waktu kali perjalanan (daftar ini disebut pula sebagai jalur Hamilton, yaitu jalur pada graph dimana setiap titiknya hanya dikunjungi satu kali). Sebuah koloni semut diciptakan dan setiap semut ditempatkan pada masing-masing titik secara merata untuk menjamin bahwa setiap titik memiliki peluang untuk menjadi titik awal dari jalur optimal yang dicari.Setiap semut selanjutnya harus melakukan tur semut, yaitu perjalanan mengunjungi semua titik pada graph tersebut (mindaputra, 2009).

Berikut adalah tahapan algoritma semut menggunakan graph dapat dilihat pada gambar 2.8:

1. Dari sarang, semut berkeliling secara acak mencari makanan sambil mencatat jarak antara titik yang dilalui.
2. Ketika sampai di sumber makanan, total jarak dari tiap titik yang ditempuh dijumlahkan untuk mendapatkan jarak dari sarang ke makanan.

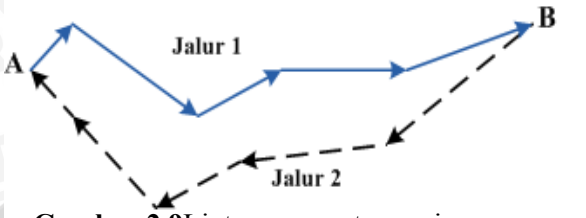


Gambar 2.8 Lintasan awal semut menuju tempat makanan

Keterangan Gambar 2.8:
A : Tempat awal koloni (sarang)
B : Tujuan koloni semut (makanan)
Jalur 1 (biru) : Lintasan yang ditempuh semut 1

Jalur 2 (hitam) : Lintasan yang ditempuh semut 2

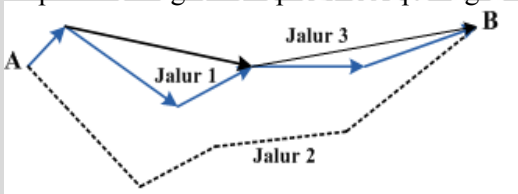
3. Ketika kembali ke sarang, sejumlah konsentrasi *pheromone*ditambahkan pada jalur yang ditempuh berdasarkan total jarak jalur tersebut. Makin kecil total jarak (atau makin optimal), maka makin banyak kadar*pheromone* yang dibubuhkan pada masing-masing garis pada jalur tersebut. Untuk lebih jelasnya dapat dilihat pada gambar 2.9.



Gambar 2.9 Lintasan semut menuju sarang

Keterangan gambar 2.9:
A : Sarang semut
B : Tempat ditemukannya makanan
Jalur 1 (biru) : Jalur yang ditempuh oleh semut 1 dengan pemberian kadar*pheromone* tinggi
Jalur 2 (hitam) : Jalur yang ditempuh oleh semut 1 dengan pemberian kadar*pheromone* tinggi

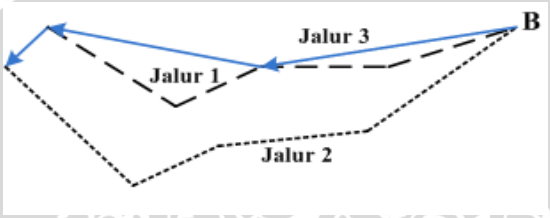
4. Untuk memilih garis mana yang harus dilalui berikutnya, digunakan sebuah rumus yang pada intinya menerapkan suatu fungsi *heuristic* untuk menghitung intensitas *pheromone*yang ditinggalkan pada suatu garis. Dapat dilihat pada gambar 2.10.



Gambar 2.10 Lintasan semut menuju makanan pada iterasi ke-2

Keterangan gambar 2.10:
A : Sarang semut
B : Tempat ditemukannya makanan
Jalur 1 (biru) : Jalur yang ditempuh oleh semut 1 dengan pemberian kadar*pheromone* tinggi
Jalur 2 (hitam) : Jalur yang ditempuh oleh semut 1 dengan pemberian kadar*pheromone* tinggi
Jalur 3 : Jalur yang ditemukan semut 2

5. Pada iterasi berikutnya, garis-garis yang mengandung*pheromone* lebih tinggi ini akan cenderung dipilih sebagai garis yang harus ditempuh berikutnya. Akibatnya, lama-kelamaan akan terlihat **Jalur Optimal**pada graph, yaitu jalur-jalur yang dibentuk oleh garis dengan kadar *pheromone* yang tinggi, yang pada akhirnya akan dipilih oleh semua multi agen semut. Dapat dilihat pada gambar 2.11



Gambar 2.11 Lintasan semut menuju sarang pada iterasi ke-2

Keterangan gambar 2.11:
A : Sarang semut
B : Tempat ditemukannya makanan
Jalur 1 (biru) : Jalur yang ditempuh oleh semut 2 dengan pemberian kadar*pheromone* rendah
Jalur 2 : Jalur yang ditempuh
Jalur 3 (biru) : Jalur yang ditempuh oleh semut 2 dengan pemberian kadar*pheromone* yang tinggi

6. Berdasarkan rangkaian kejadian diatas, maka terpilihlah **jalur optimal** yaitu jalur ke-3 dengan menggunkan metode *heuristic* menggunakan algoritma koloni semut.

(Wardy, 2007)

2.4.5 Inisialisasi Parameter Algoritma Koloni Semut

Dalam algoritma koloni semut terdapat beberapa parameter masukan sebagai inisialisasi awal untuk melakukan proses optimasi. Beberapa parameter tersebut adalah:

N : Banyak titik

$\tau(i,j)$: Inisialisasi *pheromone* awal

α : Parameter untuk mengendalikan tingkah kepentingan relatif dan jejak *pheromone*

β : Parameter untuk mengendalikan tingkah kepentingan relatif visibilitas

ρ : Evaporasi *pheromone*

NCmax : Jumlah siklus maksimum, bersifat tetap selama algoritma dijalankan.

(Kusumadewi dan Purnomo, 2005)

2.4.6 Pengisian Titik pada *tabu list*

Hasil inisialisasi pertama setiap semut harus diisikan pada *tabulist*sehingga pada awalnya tabu list terisi oleh indeks titik tertentu, yang berarti bahwa setiap *tabulist* bisa terisi indeks antara titik 1 sampai titik *n*sebagaimana hasil dari inisialisasi titik.

(Kusumadewi dan Purnomo, 2005)

2.4.7 Pemilihan Rute Perjalanan

Dalam memilih rute perjalanan (titik yang akan dikunjungi selanjutnya), semut menggunakan persamaan peluang sebagai berikut :

$$\boldsymbol{Pij(t)} = \begin{cases} \dfrac{\boldsymbol{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}}{\sum \boldsymbol{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}}; J \in U \\ 0; \text{lainnya} \end{cases} \quad (2.2)$$

dimana**:**

Pij : Peluang transisi dari titik *i* ke *j* pada saat *t*

$\tau_{ij}(t)$: Intensitas jejak *pheromone* pada garis (*i*, *j*) pada saat (t)

$\eta_{ij}(t)$: Visibilitas node *i* dan *j* pada saat *t*, $\eta_{ij}(t) = \frac{1}{d(i,j)}$

α : Parameter untuk mengendalikan tingkat kepentingan relative dari jejak *pheromone*

β : Parameter untuk mengendalikan tingkat kepentingan relatif dari visibilitas.

Setelah didapatkan peluang perpindahan dari setiap titik, kemudian dihitung nilia komulatif dari setiap titik yaitu dengan menjumlahkan peluang titik awal samapai titik yang dihitung. Nilai eluang komulatif tersebut akan dibandingkan dengan bilangan yang dibangkitkan secara acak. Jika peluang titik yang akan dituju jumlahnya sudah lebih besar dari nilai bilangan acak yang dibangkitkan maka titik tersebut akan dipilih sebagai titik yang akan dikunjungi. Titik tersebut akan disimpan sebagai titik yang telah dikunjungi oleh semut, sehingga untuk menentukan titik selanjutnya tidak akan dimasukkan kedalam proses pencarian.

(Kusumadewi dan Purnomo, 2005)

2.4.8 Perhitungan Panjang Rute Setiap Semut

Setelah koloni semut menyelesaikan perjalanannya, maka akan didapat rute dari setiap semut pada siklus tersebut. Dari rute yang ada akan dihitung panjang rute yang dilalui oleh seiap semut. Perhitungan panjang rute tersebut dilakukan setiap semut telah menyelesaikan satu siklus dan semua titik telah dikunjungi. Perhitungan ini dilakukan berdasarkan isi dari *tabulist*yang telah terisi dengan persamaan:

$$\boldsymbol{L^k = \sum_{i=1}^{N-1} T(i, i+1)} \qquad (2.3)$$

dimana :

L^k : Panjang lintasan yang dilalui oleh semut *k*

N : Banyak titik
T(i,i +1) : bobot antara titik ke i dan i +1 pada *tabu list*

Setelah seluruh nilai L^k dihitung maka akan didapatkan nilai minimum dari panjang rute secara secara keseluruhan.

2.4.9 Perhitungan Perubahan Nilai Intensitas *Pheromone* Antar Titik

Setelah semua semut melalui semua titik yang tersedia, proses berikutnya adalah melakukan perubahan nilai intensitas*pheromone*antar titik. Hal ini dilakukan kerena jumlah semut yang melewati setiap garis berbeda-beda, semakin banyak semut yang melewati suatu garis maka intensitas *pheromone*akan semakin tinggi begitu juga sebaliknya. Berikut persamaan yang digunakan untuk menghitung nilai perubahan intensitas *pheromone* antar titik:

$$\tau_{ij}(t+N) = (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t,t+N) \quad (2.4)$$

$$\Delta\tau_{ij}(t,t+N) = \begin{cases} \frac{1}{L^k N} & ;\text{Semut } k \text{ menggunakan garis } (i,j) \\ 0 & ;\text{Semut k tidak menggunakan garis } (i,j) \end{cases}$$

dimana :

$\tau_{ij}(t)$: Intensitas jejak *pheromone* pada garis (i, j) Pada saat t

$\tau_{ij}(t+N)$: Intensitas jejak *pheromone* setelah semut menyelesaikan perjalanan dalam 1 siklus

$\Delta\tau_{ij}(t,t+N)$: Intensitas jejak *pheromone* yang ditinggalkan oleh semut k pada garis (i, j) pada interval t dan t +1

L^k : panjang lintasan yang dilalui semut k

ρ : koefisien penguapan *pheromone*

N : banyak titik

(Mindaputra, 2009)

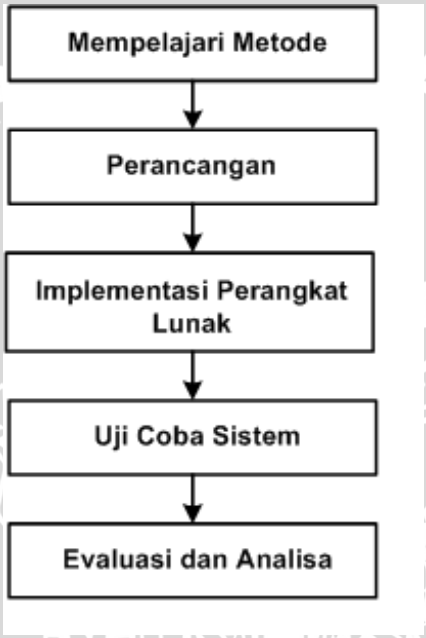
2.4.10 Pengosongan *Tabu List*

Sebelum memasuki siklus selanjutnya maka *tabu list*perlu dikosongkan terlebih dahulu. Jika jumlah siklus maksimum belum tercapai atau belum terjadi konvergensi maka algoritma diulang dari dari penyusunan rute kunjungan semut dengan nilai intensitas *pheromone* antar titik yang telah diperbarui. Proses berhenti jika siklus maksimum telah tercapai (Mindaputra, 2009).

BAB III
METODOLOGI DAN PERANCANGAN

Bab ini akan membahas mengenai perancangan perangkat lunak untuk penjadwalan mata kuliah pada jurusan matematika Universitas Brawijaya Malang. Pembahasan ini meliputi deskripsi sistem secara umum, yaitu perancangan data, perancangan proses dan perancangan antarmuka. Perancangan data terdiri dari perancangan masukan data, data proses dan data keluaran. Pada proses masukan data digunakan *database*.

Dalam penelitian ini, penulis melalui beberapa tahapan sehingga diperoleh suatu perangkat lunak penyusunan penjadwalan mata kuliah yang cepat dan akurat di jurusan matematika Universitas Brawijaya Malang.Langkah – langkah penelitian ini ditunjukkan oleh Gambar 3.1 berikut.



Gambar 3.1 Tahapan-tahapan penelitian

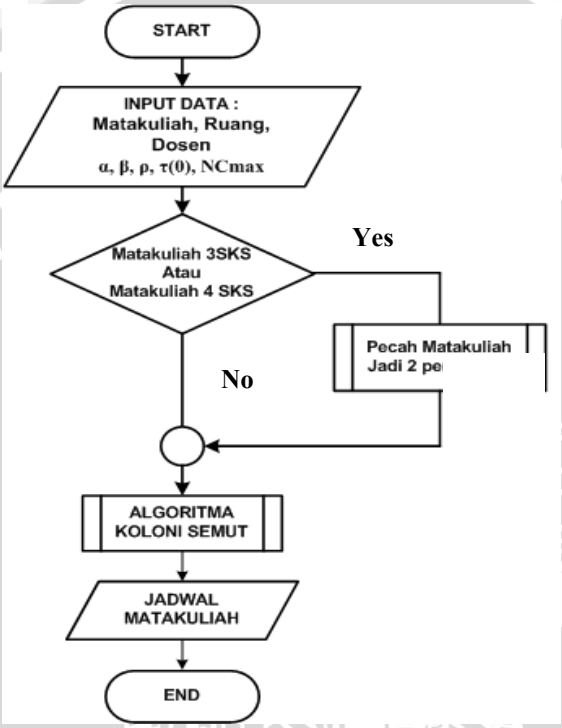
Penelitian dilakukan dengan tahapan sebagai berikut :

1. Mempelajari metode yang digunakan dari berbagai sumber seperti yang telah dijelaskan pada bab 2.

2. Merancang perangkat lunak dengan metode yang akan digunakan.
3. Membuat perangkat lunak berdasarkanperancangan yang dilakukan.
4. Uji coba penjadwalan mata kuliah menggunakan perangkat lunak yang telah dibuat.
5. Evaluasi dan analisa hasil uji coba.

3.1 Perancangan Flowchart

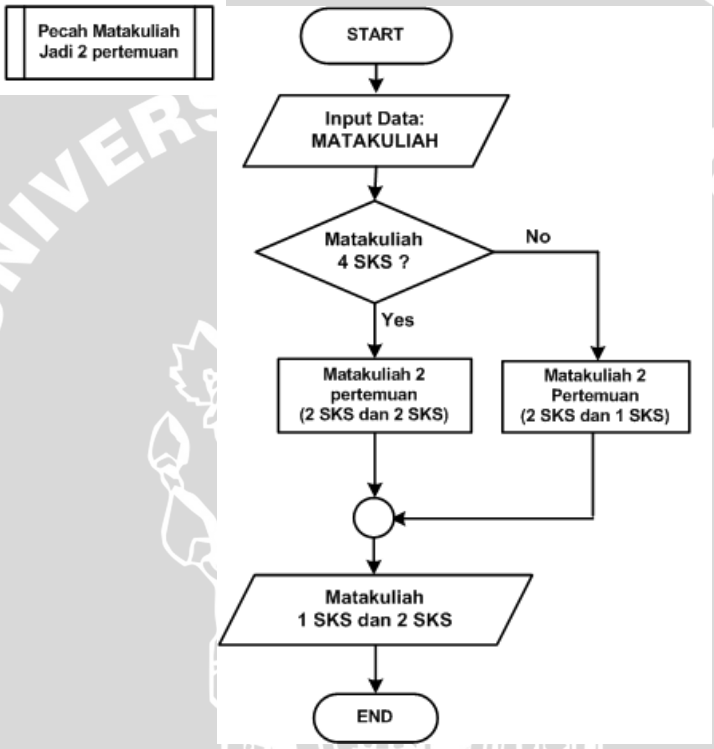
Dalam penyelesaiaan masalah penjadwalan mata kuliah dengan algoritma koloni semut, urutan proses sistem secara umum ditunjukkan dalam flowchart pada gambar 3.2.



Gambar 3. 2 Flowchart proses sistem penjadwalan kuliah

1. Langkah 1 : Input data mata kuliah, dosen pengajar dan ruangan yang akan digunakan. Dalam input mata kuliah terdapat jumlah SKS mata kuliah tersebut.
   Input nilai parameter : α, β, ρ, τ (0) dan NCmax.
2. Langkah 2 : Dilakukan pengecekan terhadap jumlah SKS mata kuliah yang di inputkan. Mata kuliah 3 SKS atau 4 SKS menjadi dua pertemuan.
3. Langkah 3 : Proses algoritma koloni semut.
4. Langkah 4 : Jadwalmata kuliah yang dihasilkan.

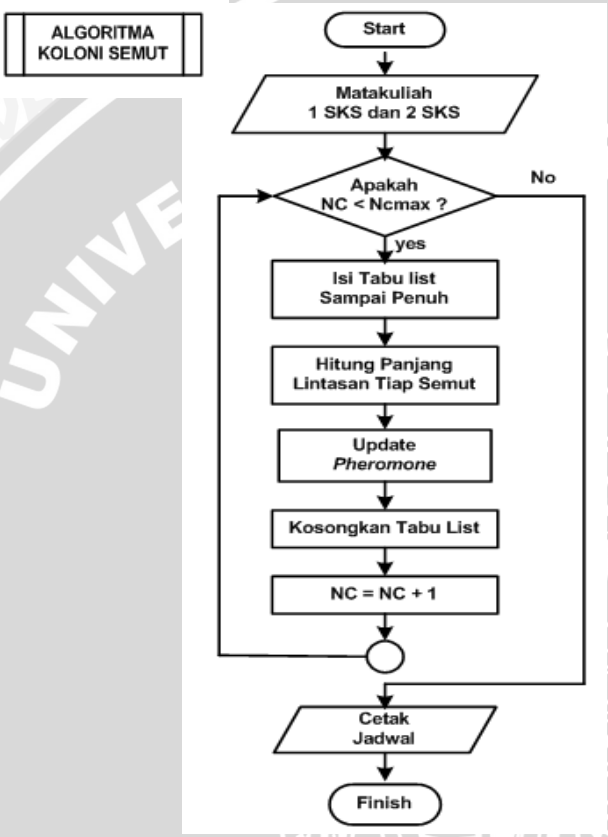
Dari flowchart gambar 3.2, proses pemecahan mata kuliah masih berupa *procedure*, sehingga bisa kita tunjukkan dalam flowchart gambar 3.3.



Gambar 3.3 Flowchart pemecahan pertemuan mata kuliah

1. Langkah 1 : Input data mata kuliah.
2. Langkah 2 :Dilakukan seleksi terhadap mata kuliahuntuk dipecah sesuai dengan jumlah SKSmata kuliah tersebut.
    - Jika mata kuliah terdiri dari 4 SKS, maka mata kuliah akan dipecah menjadi 2 SKS dan 2 SKS.
    - Jika mata kuliah terdiri dari 3 SKS, maka mata kuliah akan dipecah menjadi 2 SKS dan 1 SKS.
3. Langkah 3 : diperoleh kelompok mata kuliah 1 SKS dan 2 SKS.

Sedangkanproses algoritma koloni semut harus disesuaikan dengan aplikasi penjadwalan mata kuliah yang akan dibuat, seperti flowchart gambar 3.4.



Gambar 3.4 Algoritma koloni semut

1. Langkah 1 : Pengisian kedalam *tabu list* banyak titik yang ada. Dalam hal ini adalah kumpulah mata kuliah1 SKS dan 2 SKS yang sudah dipecah berdasarkan flowchart gambar 3.3.
2. Langkah 2 : Penyusunan rute kunjungan setiap semut ke setiap titik atau node.
3. Langkah 3 : Ulangi algoritma dari langkah 2 sehingga setiap semut telah mengunjungi semua titik (*tabu list* penuh).
4. Langkah 4 : Hitung panjang lintasan pada setiap semut.
5. Langkah 5 : Pencarian rute dengan panjang lintasan terpendek
6. Langkah 6 : Perhitungan perubahan intensitas jejak *pheromone* semut antar titik
7. Langkah 7 : perbaikan *pheromone* atau *Update*nilai perubahan intensitas jejak *pheromone* antar titik.
8. Langkah 8 : pengosongan *tabu list*.

3.2 *Constraint*

Constraint merupakan syarat yang tidak boleh dilanggar agar dapat mendapatkan solusi dari algoritma semut dalam kasus penjadwalan.Syarat ini bisa dibagi menjadi 2 kategori yaitu *soft constraint* dan *hard constraint*.*Soft constraint* berarti proses penjadwalan ini memiliki prioritas yang dilarang, tetapi masih boleh dilanggar karena masih bisa ditoleransi jika terjadi pelanggaran, sedangkan untuk*hard constraint*harus benar-benar terpenuhi sebab jika tidak dipenuhi maka solusi yang diinginkan tidak akan pernah bisa dicapai. Adapun *hard constraint* yang diterapkan dalam penjadwalan mata kuliah ini antara lain:

1. Dalam satu waktu tidak ada dosen yang mengajar di dua tempat.
2. Mata kuliah dengan semester sama, prodi yang sama, kelas yang sama tidak boleh dijadwalkan dalam hari yang sama dan waktu yang sama.
3. Ruang perkuliahan tidak boleh di pakai pada waktu yang bersamaan.

4. Mata kuliah 3 SKS dibagi jadi 2 SKS dan 1 SKS tidak boleh dijadwalkan di hari yang sama.
5. Mata kuliah 4 SKS dibagi menjadi 2 SKS dan 2 SKS tidak boleh dijadwalkan di hari yang sama.

Soft constraint ini tetap harus dipenuhi namun tidak wajib, jadi masih ada kemungkinan terjadi pelanggaran pada *soft constraint*.Hal ini terjadi apabila syarat *hard constraint* memaksakan kondisi tertentu sehingga mata kuliah yang memenuhi syarat *soft constraint* harus keluar dari persyaratan.*Soft constraint* yang diterapkan dalam penjadwalan mata kuliah ini antara lain:

1. Mata kuliah semester awal dijadwalkan lebih pagi.
2. Pecahan mata kuliah dijadwalkan satu hari setelah pecahan sebelumnya.
3. Area mengajar dosen yang berkelanjutan ditempatkan di area yang berdekatan.
4. Dosen dapat memesan waktu mengajar tertentu yang diinginkan.

3.3 Pengolahan Data Manual Algoritma Koloni Semut

Untuk mengetahui kebenaran jalannya algoritma koloni semut, maka diberikan satu kasus penjadwalan mata kuliah sederhana dengan beberapa mata kuliah yang ditawarkan. Demikian pula jam dan ruangan yang akan digunakan juga lebih sederhana. Hal ini dilakukan dengan tujuan mencari optimasi dari jalur yang terbentuk. Tabel 3.1 menunjukkan data mata kuliah yang akan dioptimalkan, namun belum dipecah berdasarkan aturan yang telah ditentukan (mata kuliah 3 SKS dan 4 SKS dipecah menjadi dua pertemuan).

Asumsi yang digunakan dalam contoh ini adalah ruangan yang tersedia hanya ada 4 ruangan dalam setiap jamnya dan waktu yang tersedia adalah 3 jam dan jumlah hari yang digunakan adalah 3 hari. Dengan beberapa asumsi diatas, akan dicari jadwal yang paling optimal.

Tabel 3.1 Tabel mata kuliah sebelum dipecah

Mata kuliah	Kelas	SKS	Semester	Dosen
Matematika Diskrit	A	2	2	Indah
Algoritma Genetika	A	3	4	Arif
Algoritma Genetika	B	3	4	Dian
Basis Data I	A	2	4	Marji
Rekayasa P. Lunak	A	3	6	Bondan
Kemanan Jaringan	A	3	6	Reza

Selanjutnya mata kuliah pada tabel 3.1 harus dipecah berdasarkan *hard constraint* yang sudah ditetapkan.Tabel 3.2 menunjukkan mata kuliah yang sudah dipecah berdasarkan jumlah SKS yang ada. Dengan adanya pemecahan mata kuliah tersebut, maka jumlah titik atau node yang akan terbentuk dalam graph lengkap G akan semakin banyak titiknya. Hal ini harus dilakukan mengingat *hard constraint* adalah aturan yang benar-benar tidak boleh dilanggar. Jika *hard constraint* dilanggar, maka jadwal yang dihasilkan nantinya tidak akan optimal.

Tabel 3.2 Tabel mata kuliah sesudah dipecah dan berkode

Kode Titik	Mata Kuliah	Kelas	SKS	Semester	Dosen
1	Matematika Diskrit	A	2	2	Indah Y
2	Algoritma Genetika	A	2	4	Arif
3	Algoritma Genetika	A	1	4	Arif
4	Algoritma Genetika	B	2	4	Dian
5	Algoritma Genetika	B	1	4	Dian
6	Basis Data I	A	2	4	Marji
7	Rekayasa P. Lunak	A	2	6	Bondan
8	Rekayasa P. Lunak	A	1	6	Bondan
9	Kemanan Jaringan	A	2	6	Reza
10	Kemanan Jaringan	A	1	6	Reza

Setelah didapatkan titik-titik pada graph (10 titik yang mewakili masing-masing mata kuliah), setiap titik pada graph tersebut akan dihubungkan, sehingga diperoleh graph lengkap G. selanjutnya pada masing-masing garis akan diberikan bobot. Bobot tersebut merupakan nilai penalti dari titik *i* dan titik *j*. nilai penalti tersebut diformulasikan sebagai :

$$d(i,j) = \begin{cases} 3 \text{ ; jika aturan 1 atau 2 dilanggar} \\ 6 \text{ ; jika aturan 1 dan 2 dilanggar} \\ 0 \text{ ; lainnya} \end{cases}$$

Aturan-aturan yang dimaksud adalah :

1. Mata kuliah dengan semester yang sama tidak boleh dijadwalkan dalam waktu yang sama kecuali kelas mata kuliah atau prodi mata kuliah berbeda.
2. Dosen tidak boleh mengajar lebih dari satu mata kuliah dalam waktu yang sama.

Sebagai contoh kejadian kres pada dosen dan mahasiswa diberikan penalti 6, sedangkan kres pada dosen atau mahasiswa diberikan penalti 3. Tabel 3.3 menunjukkan nilai atau bobot graph yang terbentuk dari susunan mata kuliah pada tabel 3.2 yang terdiri dari 10 titik dimana antar titik saling terhubung dan memiliki bobot tertentu sesuai dengan aturan penalti.

Tabel 3.3 Tabel Bobot Antar Titik graph

Titik	1	2	3	4	5	6	7	8	9	10
1	-	0	0	0	0	0	0	0	0	0
2	0	-	6	6	6	3	0	0	0	0
3	0	6	-	6	6	3	0	0	0	0
4	0	6	6	-	6	3	0	0	0	0
5	0	6	6	6	-	3	0	0	0	0
6	0	3	3	3	3	-	0	0	0	0
7	0	0	0	0	0	0	-	6	3	3
8	0	0	0	0	0	0	6	-	3	3
9	0	0	0	0	0	0	3	3	-	6
10	0	0	0	0	0	0	3	3	6	-

Selanjutnya dari tabel 3.3 akan dibentuk nilai visibilitas antar titik ($\acute{\eta}_{ij}$) dari masing-masing titik yang ada seperti tabel 3.4 berikut:

Tabel 3.4 Visibilitas antar titik ($\acute{\eta}_{ij}$)

$\acute{\eta}_{ij}$	1	2	3	4	5	6	7	8	9	10
1	0	1	1	1	1	1	1	1	1	1
2	1	0	0.1429	0.1429	0.1429	0.25	1	1	1	1
3	1	0.1429	0	0.1429	0.1429	0.25	1	1	1	1
4	1	0.1429	0.1429	0	0.1429	0.25	1	1	1	1
5	1	0.1429	0.1429	0.1429	0	0.25	1	1	1	1
6	1	0.25	0.25	0.25	0.25	0	1	1	1	1
7	1	1	1	1	1	1	0	0.1429	0.25	0.25
8	1	1	1	1	1	1	0.1429	0	0.25	0.25
9	1	1	1	1	1	1	0.25	0.25	0	0.1429
10	1	1	1	1	1	1	0.25	0.25	0.1429	0

3.3.1 Inisialisasi Harga Parameter

Dalam kasus ini, inisialisasi harga parameter digunakan untuk langkah selanjutnya. Parameter-parameter tersebut adalah:

1. Jumlah titik (node) = 10
2. Jarak antar titik diberikan dalam bentuk tabel, yaitu pada tabel 3.3
3. Nilai $\alpha = 1$, $\beta = 1$, $\rho = 1$, $m = N = 10$

4. Nilai *pheromone* awal τ(0) adalah 0.01. penetapan nilai *pheromone*awal ini dimaksudkan agar tiap-tiap titik memiliki nilai ketertarikan untuk dikunjungi.

Tabel 3. 5 Intensitas jejak *pheromone* awal τ(0)

τ(0)	1	2	3	4	5	6	7	8	9	10
1	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
2	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
4	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
5	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
6	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
7	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
8	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
9	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
10	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01

3.3.2 Pengisian ke dalam *Tabu List*

Tabu list digunakan untuk menyimpan urutan titik yang telah dikunjungi setiap semut. Pada awalnya *tabu list* setiap semut kosong, kemudian setiap kali setiap semut berkunjung ke satu titik, elemen titik bertambah satu. Demikian seterusnya hingga semua node yang ada telah dikunjungi dan disimpan dalam *tabu list* (*tabu list* penuh). Selain digunakan untuk menyimpan daftar urutan titik, tabulist juga berfungsi untuk mengetahui titik-titik yang belum dikunjungi.

Titik-titik pertama setiap semut hasil inisialisasi pada langkah pertama harus diisikan sebagai elemen pertama *tabu list*. Jadi setiap $tabu_k$ (1) bisa berisi indeks titik antara 1 sampai n titik sebagaimana hasil inisialisasi pada langkah pertama. Maka, elemen *tabu list* semut pertama akan terisi indeks*tabu list* semut pertama dapat dilihat pada tabel 3.6:

Tabel 3. 6*Tabu list* awal

M	Lintasan
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10

3.4 Penelusuran Rute Kunjungan Setiap Semut ke Setiap Titik

Setelah setiap semut terdistribusi ke sejumlah titik, koloni semut akan mulai melakukan perjalanan dari titik asal ke titik selanjutnya sebagai titik tujuan. Dalam menentukan titik tujuan setiap semut akan menghitung probabilitas semua kemungkinan titik yang berhubungan dengan titik dimana dia berada. Perhitungan ini menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

1. Siklus ke-1 ; Semut -1
   - t = 1
   - Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} {\tau_{(1,j)}}^{\alpha} {\eta_{(1,j)}}^{\beta} = {\tau_{(1,1)}}^{1}{\eta_{(1,1)}}^{1} + {\tau_{(1,2)}}^{1}{\eta_{(1,2)}}^{1} + .. {\tau_{(1,10)}}^{1}{\eta_{(1,10)}}^{1}$$

$$\sum_{j=1}^{10} {\tau_{(1,j)}}^{\alpha} {\eta_{(1,j)}}^{\beta} = 0 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1$$

$$\sum_{j=1}^{10} {\tau_{(1,j)}}^{\alpha} {\eta_{(1,j)}}^{\beta} = 0.09$$

Setelah didapat nilai komulatif intensitas jejak *pheronome* selanjutnya menghitung peluang antar titik.

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-1:

$$P_{(1,1)} = 0$$

$$P_{(1,2)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,3)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,4)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,5)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,6)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,7)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,8)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,9)} = \frac{0.01}{0.09} = 0.11111$$

$$P_{(1,10)} = \frac{0.01}{0.09} = 0.11111$$

- Tabel 3.7 menunjukkan tabel peluang perpindahan semut -1 ke titik yang lain.

Tabel 3.7Peluang perpindahan semut-1

$P_{(1,j)}$	Komulatif
$P_{(1,1)} = 0$	0
$P_{(1,2)} = 0.11111$	0.11111
$P_{(1,3)} = 0.11111$	0.22222
$P_{(1,4)} = 0.11111$	0.33333
$P_{(1,5)} = 0.11111$	0.44444
$P_{(1,6)} = 0.11111$	0.55555
$P_{(1,7)} = 0.11111$	0.66666
$P_{(1,8)} = 0.11111$	0.77777
$P_{(1,9)} = 0.11111$	0.88888
$P_{(1,10)} = 0.11111$	0.99999

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.30841**
- Titik berikutnya yang dipilih : **4**

2. Siklus ke-1 ; Semut -2

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone* :

$$\sum_{j=1}^{10} \tau_{(2,j)}{}^{\alpha}\, \eta_{(2,j)}{}^{\beta} = \tau_{(2,1)}{}^{1}\eta_{(2,1)}{}^{1} + \tau_{(2,2)}{}^{1}\eta_{(2,2)}{}^{1} + .. \tau_{(2,10)}{}^{1}\eta_{(2,10)}{}^{1}$$

$$\sum_{j=1}^{10} \tau_{(2,j)}{}^{\alpha}\, \eta_{(2,j)}{}^{\beta} = (0.01)^1 + 0 + (0.01)^1(0.1429)^1 +$$

$$(0.01)^1(0.1429)^1 + (0.01)^1(0.1429)^1 + (0.01)^1(0.25)^1$$

$$(0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1$$

$$\sum_{j=1}^{10} \tau_{(2,j)}{}^{\alpha}\, \eta_{(2,j)}{}^{\beta} = 0.05679$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-2:

$$P_{(2,1)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(2,2)} = 0$$
$$P_{(2,3)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(2,4)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(2,5)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(2,6)} = \frac{0.0025}{0.05679} = 0.04402$$
$$P_{(2,7)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(2,8)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(2,9)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(2,10)} = \frac{0.01}{0.05679} = 0.17609$$

- Tabel 3.8 menunjukkan tabel peluang perpindahan semut -2 ke titik yang lain.

Tabel 3. 8Peluang perpindahan semut-2

$P_{(2,j)}$	Komulatif
$P_{(2,1)}$ = 0.17609	0.17609
$P_{(2,2)}$= 0	0.17609
$P_{(2,3)}$= 0.02516	0.20125
$P_{(2,4)}$= 0.02516	0.22641
$P_{(2,5)}$ = 0.02516	0.25157
$P_{(2,6)}$ = 0.04402	0.29559
$P_{(2,7)}$ = 0.17609	0.47168
$P_{(2,8)}$ = 0.17609	0.64777
$P_{(2,9)}$= 0.17609	0.82386
$P_{(2,10)}$ = 0.17609	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilanagan acak : **0.24527**
- Titik berikutnya yang dipilih : **5**

3. Siklus ke-1 ; Semut -3

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} \tau_{(3,j)}{}^{\alpha}\, \eta_{(3,j)}{}^{\beta} = \tau_{(3,1)}{}^{1}\eta_{(3,1)}{}^{1} + \tau_{(3,2)}{}^{1}\eta_{(3,2)}{}^{1} + .. \tau_{(3,10)}{}^{1}\eta_{(3,10)}{}^{1}$$

$$\sum_{j=1}^{10} \tau_{(3,j)}{}^{\alpha}\, \eta_{(3,j)}{}^{\beta} = (0.01)^1 + (0.01)^1(0.1429)^1 + 0 +$$

$$(0.01)^1(0.1429)^1 + (0.01)^1(0.1429)^1 + (0.01)^1(0.25)^1$$

$$(0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1$$

$$\sum_{j=1}^{10} \tau_{(3,j)}{}^{\alpha}\, \eta_{(3,j)}{}^{\beta} = 0.05679$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-3:

$$P_{(3,1)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(3,2)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(3,3)} = 0$$
$$P_{(3,4)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(3,5)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(3,6)} = \frac{0.0025}{0.05679} = 0.04402$$
$$P_{(3,7)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(3,8)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(3,9)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(3,10)} = \frac{0.01}{0.05679} = 0.17609$$

- Tabel 3.9 menunjukkan tabel peluang perpindahan semut -3 ke titik yang lain.

Tabel 3.9 Peluang perpindahan semut-3

$P_{(3,j)}$	Komulatif
$P_{(3,1)}$ = 0.17609	0.17609
$P_{(3,2)}$= 0.02516	0.20125
$P_{(3,3)}$= 0	0.20125
$P_{(3,4)}$= 0.02516	0.22641
$P_{(3,5)}$ = 0.02516	0.25157
$P_{(3,6)}$ = 0.04402	0.29559
$P_{(3,7)}$ = 0.17609	0.47168
$P_{(3,8)}$ = 0.17609	0.64777
$P_{(3,9)}$= 0.17609	0.82386
$P_{(3,10)}$ = 0.17609	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.86863**
- Titik berikutnya yang dipilih : **10**

4. Siklus ke-1 ; Semut -4
   - t = 1
   - Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} \tau_{(4,j)}{}^{\alpha}\, \eta_{(4,j)}{}^{\beta} = \tau_{(4,1)}{}^{1}\eta_{(4,1)}{}^{1} + \tau_{(4,2)}{}^{1}\eta_{(4,2)}{}^{1} + .. \tau_{(4,10)}{}^{1}\eta_{(4,10)}{}^{1}$$

$$(0.1429)^1 + 0 + (0.01)^1(0.1429)^1 + (0.01)^1(0.25)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1$$

$$\sum_{j=1}^{10} \tau_{(4,j)}{}^{\alpha}\, \eta_{(4,j)}{}^{\beta} = 0.05679$$

   - Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-4:

$$P_{(4,1)} = \frac{0.01}{0.05679} = 0.17609$$

$$P_{(4,2)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(4,3)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(4,4)} = 0$$
$$P_{(4,5)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(4,6)} = \frac{0.0025}{0.05679} = 0.04402$$
$$P_{(4,7)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(4,8)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(4,9)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(4,10)} = \frac{0.01}{0.05679} = 0.17609$$

- Tabel 3.10 menunjukkan tabel peluang perpindahan semut -4 ke titik yang lain.

Tabel 3.10 Peluang perpindahan semut-4

$P_{(4,j)}$	Komulatif
$P_{(4,1)}$ = 0.17609	0.17609
$P_{(4,2)}$= 0.02516	0.20125
$P_{(4,3)}$= 0.02516	0.22641
$P_{(4,4)}$= 0	0.22641
$P_{(4,5)}$ = 0.02516	0.25157
$P_{(4,6)}$ = 0.04402	0.29559
$P_{(4,7)}$ = 0.17609	0.47168
$P_{(4,8)}$ = 0.17609	0.64777
$P_{(4,9)}$= 0.17609	0.82386
$P_{(4,10)}$ = 0.17609	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.63582**

- Titik berikutnya yang dipilih : **8**

5. Siklus ke-1 ; Semut -5

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} \tau_{(5,j)}{}^{\alpha}\, \eta_{(5,j)}{}^{\beta} = \tau_{(5,1)}{}^{1}\eta_{(5,1)}{}^{1} + \tau_{(5,2)}{}^{1}\eta_{(5,2)}{}^{1} + .. \tau_{(5,10)}{}^{1}\eta_{(5,10)}{}^{1}$$

$$\sum_{j=1}^{10} \tau_{(5,j)}{}^{\alpha}\, \eta_{(5,j)}{}^{\beta} = (0.01)^1 + (0.01)^1(0.1429)^1 + (0.01)^1. (0.1429)^1 + (0.01)^1(0.1429)^1 + 0 + (0.01)^1(0.25)^1 + + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1$$

$$\sum_{j=1}^{10} \tau_{(5,j)}{}^{\alpha}\, \eta_{(5,j)}{}^{\beta} = 0.05679$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-5:

$$P_{(5,1)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(5,2)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(5,3)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(5,4)} = \frac{0.001429}{0.05679} = 0.02516$$
$$P_{(5,5)} = 0$$
$$P_{(5,6)} = \frac{0.0025}{0.05679} = 0.04402$$
$$P_{(5,7)} = \frac{0.01}{0.05679} = 0.17609$$
$$P_{(5,8)} = \frac{0.01}{0.05679} = 0.17609$$

$$P_{(5,9)} = \frac{0.01}{0.05679} = 0.17609$$

$$P_{(5,10)} = \frac{0.01}{0.05679} = 0.17609$$

- Tabel 3.11 menunjukkan tabel peluang perpindahan semut -5 ke titik yang lain.

Tabel 3.11 Peluang perpindahan semut-5

$P_{(5,j)}$	Komulatif
$P_{(5,1)} = 0.17609$	0.17609
$P_{(5,2)} = 0.02516$	0.20125
$P_{(5,3)} = 0.02516$	0.22641
$P_{(5,4)} = 0.02516$	0.25157
$P_{(5,5)} = 0$	0.25157
$P_{(5,6)} = 0.04402$	0.29559
$P_{(5,7)} = 0.17609$	0.47168
$P_{(5,8)} = 0.17609$	0.64777
$P_{(5,9)} = 0.17609$	0.82386
$P_{(5,10)} = 0.17609$	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.21582**
- Titik berikutnya yang dipilih : **3**

6. Siklus ke-1 ; Semut -6

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} \tau_{(6,j)}{}^{\alpha}\, \eta_{(6,j)}{}^{\beta} = \tau_{(6,1)}{}^{1}\eta_{(6,1)}{}^{1} + \tau_{(6,2)}{}^{1}\eta_{(6,2)}{}^{1} + .. \tau_{(6,10)}{}^{1}\eta_{(6,10)}{}^{1}$$

$$\sum_{j=1}^{10} \tau_{(6,j)}{}^{\alpha}\, \eta_{(6,j)}{}^{\beta} = (0.01)^1 + (0.01)^1(0.25)^1 + (0.01)^1(0.25)^1$$

$$(0.01)^1(0.25)^1 + (0.01)^1(0.25)^1 + 0 + (0.01)^1 + (0.01)^1 + (0.01)^1$$

$$\sum_{j=1}^{10} \tau_{(6,j)}{}^{\alpha}\, \eta_{(6,j)}{}^{\beta} = 0.06000$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-6:

$$P_{(6,1)} = \frac{0.01}{0.06000} = 0.16667$$

$$P_{(6,2)} = \frac{0.0025}{0.06000} = 0.04167$$

$$P_{(6,3)} = \frac{0.0025}{0.06000} = 0.04167$$

$$P_{(6,4)} = \frac{0.0025}{0.06000} = 0.04167$$

$$P_{(6,5)} = \frac{0.0025}{0.06000} = 0.04402$$

$$P_{(6,6)} = 0$$

$$P_{(6,7)} = \frac{0.01}{0.06000} = 0.16667$$

$$P_{(6,8)} = \frac{0.01}{0.06000} = 0.16667$$

$$P_{(6,9)} = \frac{0.01}{0.06000} = 0.16667$$

$$P_{(6,10)} = \frac{0.01}{0.06000} = 0.16667$$

- Tabel 3.12 menunjukkan tabel peluang perpindahan semut -6 ke titik yang lain.

Tabel 3. 12 Peluang perpindahan semut-6

$P_{(6,j)}$	Komulatif
$P_{(6,1)}$ = 0.16667	0.16667
$P_{(6,2)}$= 0.04167	0.20834
$P_{(6,3)}$= 0.04167	0.25001
$P_{(6,4)}$= 0.04167	0.29168
$P_{(6,5)}$ = 0.04167	0.33335
$P_{(6,6)}$ = 0	0.33335
$P_{(6,7)}$ = 0.16667	0.50002
$P_{(6,8)}$ = 0.16667	0.66669
$P_{(6,9)}$= 0.16667	0.83336
$P_{(6,10)}$ = 0.16667	1.00000

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.20150**
- Titik berikutnya yang dipilih : **2**

7. Siklus ke-1 ; Semut -7

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} \tau_{(7,j)}{}^{\alpha}\, \eta_{(7,j)}{}^{\beta} = \tau_{(7,1)}{}^{1}\eta_{(7,1)}{}^{1} + \tau_{(7,2)}{}^{1}\eta_{(7,2)}{}^{1} + .. \tau_{(7,10)}{}^{1}\eta_{(7,10)}{}^{1}$$

$$\sum_{j=1}^{10} \tau_{(7,j)}{}^{\alpha}\, \eta_{(7,j)}{}^{\beta} = (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 +$$

$$(0.01)^1 + (0.01)^1 + 0 + (0.01)^1(0.1429)^1 + \\ +(0.01)^1(0.25)^1 + (0.01)^1(0.25)^1$$

$$\sum_{j=1}^{10} \tau_{(7,j)}{}^{\alpha}\, \eta_{(7,j)}{}^{\beta} = 0.06643$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-7:

$$P_{(7,1)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(7,2)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(7,3)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(7,4)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(7,5)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(7,6)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(7,7)} = 0$$
$$P_{(7,8)} = \frac{0.001429}{0.06643} = 0.02151$$
$$P_{(7,9)} = \frac{0.0025}{0.06643} = 0.03763$$
$$P_{(7,10)} = \frac{0.0025}{0.06643} = 0.03763$$

- Tabel 3.13 menunjukkan tabel peluang perpindahan semut -7 ke titik yang lain.

Tabel 3. 13 Peluang perpindahan semut-7

$P_{(7,j)}$	Komulatif
$P_{(7,1)}$ = 0.15053	0.15053
$P_{(7,2)}$= 0.15053	0.30106
$P_{(7,3)}$= 0.15053	0.45159
$P_{(7,4)}$= 0.15053	0.60212
$P_{(7,5)}$ = 0.15053	0.75265
$P_{(7,6)}$ = 0.15053	0.90318
$P_{(7,7)}$ = 0	0.90318
$P_{(7,8)}$ = 0.02151	0.92469
$P_{(7,9)}$= 0.03763	0.96232
$P_{(7,10)}$ = 0.03763	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.95888**
- Titik berikutnya yang dipilih : **9**

8. Siklus ke-1 ; Semut -8

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} \tau_{(8,j)}{}^{\alpha}\, \eta_{(8,j)}{}^{\beta} = \tau_{(8,1)}{}^{1}\eta_{(8,1)}{}^{1} + \tau_{(8,2)}{}^{1}\eta_{(8,2)}{}^{1} + .. \tau_{(8,10)}{}^{1}\eta_{(8,10)}{}^{1}$$

$$\sum_{j=1}^{10} \tau_{(8,j)}{}^{\alpha}\, \eta_{(8,j)}{}^{\beta} = (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1(0.1429)^1 + 0 + +(0.01)^1(0.25)^1 + (0.01)^1(0.25)^1$$

$$\sum_{j=1}^{10} \tau_{(8,j)}{}^{\alpha}\, \eta_{(8,j)}{}^{\beta} = 0.06643$$

Setelah mendapatkan nilai komulatif dari intensitas jejak *pheromone* maka akan dihitung probabilitasnya.

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-8:

$$P_{(8,1)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(8,2)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(8,3)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(8,4)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(8,5)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(8,6)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(8,7)} = \frac{0.001429}{0.06643} = 0.02151$$
$$P_{(8,8)} = 0$$
$$P_{(8,9)} = \frac{0.0025}{0.06643} = 0.03763$$
$$P_{(8,10)} = \frac{0.0025}{0.06643} = 0.03763$$

- Tabel 3.14 menunjukkan tabel peluang perpindahan semut -8 ke titik yang lain.

Tabel 3.14 Peluang perpindahan semut-8

$P_{(8,j)}$	Komulatif
$P_{(8,1)}$ = 0.15053	0.15053
$P_{(8,2)}$= 0.15053	0.30106
$P_{(8,3)}$= 0.15053	0.45159
$P_{(8,4)}$= 0.15053	0.60212
$P_{(8,5)}$ = 0.15053	0.75265
$P_{(8,6)}$ = 0.15053	0.90318
$P_{(8,7)}$ = 0.02151	0.92469
$P_{(8,8)}$ = 0	0.92469
$P_{(8,9)}$= 0.03763	0.96232
$P_{(8,10)}$ = 0.03763	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.90025**
- Titik berikutnya yang dipilih : **6**

9. Siklus ke-1 ; Semut -9

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} \tau_{(9,j)}{}^{\alpha}\, \eta_{(9,j)}{}^{\beta} = \tau_{(9,1)}{}^{1}\eta_{(9,1)}{}^{1} + \tau_{(9,2)}{}^{1}\eta_{(9,2)}{}^{1} + .. \tau_{(9,10)}{}^{1}\eta_{(9,10)}{}^{1}$$

$$\sum_{j=1}^{10} \tau_{(9,j)}{}^{\alpha}\, \eta_{(9,j)}{}^{\beta} = (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1(0.25)^1 + (0.01)^1(0.25)^1 + 0 + (0.01)^1(0.1429)^1$$

$$\sum_{j=1}^{10} \tau_{(9,j)}{}^{\alpha}\, \eta_{(9,j)}{}^{\beta} = 0.06643$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-9:

$$P_{(9,1)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(9,2)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(9,3)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(9,4)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(9,5)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(9,6)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(9,7)} = \frac{0.0025}{0.06643} = 0.03763$$
$$P_{(9,8)} = \frac{0.0025}{0.06643} = 0.03763$$
$$P_{(9,9)} = 0$$
$$P_{(9,10)} = \frac{0.001429}{0.06643} = 0.02151$$

Tabel 3.15 menunjukkan tabel peluang perpindahan semut -9 ke titik yang lain.

Tabel 3.15 Peluang perpindahan semut-9

$P_{(9,j)}$	Komulatif
$P_{(9,1)}$ = 0.15053	0.15053
$P_{(9,2)}$= 0.15053	0.30106
$P_{(9,3)}$= 0.15053	0.45159
$P_{(9,4)}$= 0.15053	0.60212
$P_{(9,5)}$ = 0.15053	0.75265
$P_{(9,6)}$ = 0.15053	0.90318
$P_{(9,7)}$ = 0.03763	0.94081
$P_{(9,8)}$ = 0.03763	0.97844
$P_{(9,9)}$= 0	0.97844
$P_{(9,10)}$ = 0.02151	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.93350**
- Titik berikutnya yang dipilih : **7**

10.Siklus ke-1 ; Semut -10

Perhitungan untuk komulatif jejak *pheromone* menggunakan persamaan 2.2.urutan perhitungannya sebagai berikut:

- t = 1
- Hitung komulatif intensitas jejak *pheromone*:

$$\sum_{j=1}^{10} {\tau_{(10,j)}}^{\alpha} {\eta_{(10,j)}}^{\beta} = {\tau_{(10,1)}}^{1}{\eta_{(10,1)}}^{1} + {\tau_{(10,2)}}^{1}{\eta_{(10,2)}}^{1} + {\tau_{(10,3)}}^{1}{\eta_{(10,3)}}^{1} + \cdots + {\tau_{(10,10)}}^{1}{\eta_{(10,10)}}^{1}$$

$$\sum_{j=1}^{10} {\tau_{(10,j)}}^{\alpha} {\eta_{(10,j)}}^{\beta} = (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1 + (0.01)^1(0.25)^1 + (0.01)^1(0.25)^1 + (0.01)^1(0.1429)^1 + 0$$

$$\sum_{j=1}^{10} {\tau_{(10,j)}}^{\alpha} {\eta_{(10,j)}}^{\beta} = 0.06643$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-10:

$$P_{(10,1)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(10,2)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(10,3)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(10,4)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(10,5)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(10,6)} = \frac{0.01}{0.06643} = 0.15053$$
$$P_{(10,7)} = \frac{0.0025}{0.06643} = 0.03763$$
$$P_{(10,8)} = \frac{0.0025}{0.06643} = 0.03763$$
$$P_{(10,9)} = \frac{0.001429}{0.06643} = 0.02151$$
$$P_{(10,10)} = 0$$

Tabel 3.16 menunjukkan tabel peluang perpindahan semut -10 ke titik yang lain.

Tabel 3.16Peluang perpindahan semut-10

$P_{(10, j)}$	Komulatif
$P_{(10, 1)}$ = 0.15053	0.15053
$P_{(10, 2)}$= 0.15053	0.30106
$P_{(10, 3)}$= 0.15053	0.45159
$P_{(10, 4)}$= 0.15053	0.60212
$P_{(10, 5)}$ = 0.15053	0.75265
$P_{(10, 6)}$ = 0.15053	0.90318
$P_{(10, 7)}$ = 0.03763	0.94081
$P_{(10, 8)}$ = 0.03763	0.97844
$P_{(10, 9)}$= 0.02151	0.99995
$P_{(10, 10)}$ = 0	0.99995

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.10216**
- Titik berikutnya yang dipilih : **1**

Dari perhitungan untuk nilai t = 1 diatas, maka diperoleh *tabu list* sementara untuk t = 1 dari masing-masing titik, yaitu:

Tabel 3.17*Tabu list* sementara saat t = 1

M	Lintasan
1	1 – 4
2	2 – 5
3	3 – 10
4	4 – 8
5	5 – 3
6	6 – 2
7	7 – 9
8	8 – 6
9	9 – 7
10	10 – 1

Karena setiap semut sudah melakukan kunjungan ke masing-masing titik yang akan dikunjungi berikutnya, artinya semut sudah samadengan titik yang ada yaitu 10, maka siklus akan menuju ke t=2.

Langkah pertama yang harus dilakukan adalah melakukan pembaharuan nilai intensitas jejak *pheromone*.

Berikut perhitungan perubahan intensitas jejak antar titik.

- **Bobot 0**

$$\Delta\tau_{(i,j)} = \frac{1}{(0+1)10} = 0.1$$
$$\tau_{(i,j)} = (1-\rho)\tau_{(i,j)} + \rho\Delta\tau_{(i,j)}$$
$$\tau_{(i,j)} = (1-1)(0.01) + 1(0.1) = 0.1$$

- **Bobot 3**

$$\Delta\tau_{(i,j)} = \frac{1}{(3+1)10} = 0.025$$
$$\tau_{(i,j)} = (1-\rho)\tau_{(i,j)} + \rho\Delta\tau_{(i,j)}$$
$$\tau_{(i,j)} = (1-1)(0.01) + 1(0.025) = 0.025$$

- **Bobot 6**

$$\Delta\tau_{(i,j)} = \frac{1}{(6+1)10} = 0.01429$$
$$\tau_{(i,j)} = (1-\rho)\tau_{(i,j)} + \rho\Delta\tau_{(i,j)}$$
$$\tau_{(i,j)} = (1-1)(0.01) + 1(0.01429) = 0.01429$$

Perhitungan diatas dilakukan dengan menghitung perubahan intensitas jejak dengan bobot 0, 3 dan 6 karena ketiga nilai itulah yang merupakan bobot antar titik sesuai dengan aturan penalti yang ditetapkan, tidak ada nilai yang lain.

Dengan demikian perubahaan intensitas jejak *pheromone* untuk siklus yang pertama dapat dilihat pada tabel 3.18.

Tabel 3. 18 Intensitas jejak *pheromone* siklus τ(1)

τ(1)	1	2	3	4	5	6	7	8	9	10
1	0.01	0.01	0.01	**0.1**	0.01	0.01	0.01	0.01	0.01	**0.1**
2	0.01	0.01	0.01	0.01	**0.01429**	**0.025**	0.01	0.01	0.01	0.01
3	0.01	0.01	0.01	0.01	**0.01429**	0.01	0.01	0.01	0.01	**0.1**
4	**0.1**	0.01	0.01	0.01	0.01	0.01	0.01	**0.1**	0.01	0.01
5	0.01	**0.01429**	**0.01429**	0.01	0.01	0.01	0.01	0.01	0.01	0.01
6	0.01	**0.025**	0.01	0.01	0.01	0.01	0.01	**0.1**	0.01	0.01
7	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	**0.025**	0.01
8	0.01	0.01	0.01	**0.1**	0.01	**0.1**	0.01	0.01	0.01	0.01
9	0.01	0.01	0.01	0.01	0.01	0.01	**0.025**	0.01	0.01	0.01
10	**0.1**	0.01	**0.1**	0.01	0.01	0.01	0.01	0.01	0.01	0.01

Untuk t = 2 selanjutnya dipilih dengan perhitungan yang sama dengan t = 1, namun perhitungan dimulai dengan melihat titik terakhir yang dikunjungi oleh semut tersebut. Sebagai contoh, akan kita hitung semut-4 akan berjalan ke titik berapa setelah menuju titik 8. Berikut perhitungannya.

- Semut ke-4
- t = 2
- Hitung komulatif intensitas jejak *pheromone*sesuai dengan persamaan 2.2:

$$\sum_{j=1}^{10} {\tau_{(8,j)}}^{\alpha} {\eta_{(8,j)}}^{\beta} = {\tau_{(8,1)}}^{1}{\eta_{(8,1)}}^{1} + {\tau_{(8,2)}}^{1}{\eta_{(8,2)}}^{1} + \cdots$$
$$ {\tau_{(8,10)}}^{1}{\eta_{(8,10)}}^{1}$$

$$\sum_{j=1}^{10} {\tau_{(8,j)}}^{\alpha} {\eta_{(8,j)}}^{\beta} = (0.01)^1 + (0.01)^1 + (0.01)^1 + (\mathbf{0.1})^{\mathbf{1}} +$$
$$(0.01)^1 + (\mathbf{0.1})^{\mathbf{1}} + (0.01)^1(0.1429)^1 + 0 +$$
$$+(0.01)^1(0.25)^1 + (0.01)^1(0.25)^1$$

$$\sum_{j=1}^{10} {\tau_{(8,j)}}^{\alpha} {\eta_{(8,j)}}^{\beta} = 0.24643$$

- Hitung peluang titik yang akan dikunjungi selanjutnya oleh semut-8:

$$P_{(8,1)} = \frac{0.01}{0.24643} = 0.04058$$

$$P_{(8,2)} = \frac{0.01}{0.24643} = 0.04058$$
$$P_{(8,3)} = \frac{0.01}{0.24643} = 0.04058$$
$$P_{(8,4)} = \frac{0.1}{0.24643} = 0.40579$$
$$P_{(8,5)} = \frac{0.01}{0.24643} = 0.04058$$
$$P_{(8,6)} = \frac{0.1}{0.24643} = 0.40579$$
$$P_{(8,7)} = \frac{0.001429}{0.24643} = 0.00570$$
$$P_{(8,8)} = 0$$
$$P_{(8,9)} = \frac{0.0025}{0.24643} = 0.01014$$
$$P_{(8,10)} = \frac{0.0025}{0.24643} = 0.01014$$

- Tabel 3.19 menunjukkan tabel peluang perpindahan semut -4 ke titik yang lain.

Tabel 3. 19 Peluang perpindahan semut-4

$P_{(8,j)}$	Komulatif
$P_{(8,1)}$ (2)= 0.04058	0.04058
$P_{(8,2)}$ (2)= 0.04058	0.08116
$P_{(8,3)}$ (2)= 0.04058	0.12174
$P_{(8,4)}$ (2)= 0.40579	0.52753
$P_{(8,5)}$ (2)= 0.04058	0.56811
$P_{(8,6)}$ (2)= 0.40579	0.97390
$P_{(8,7)}$ (2)= 0.00570	0.97960
$P_{(8,8)}$ (2)= 0	0.97960
$P_{(8,9)}$ (2)= 0.01014	0.98974
$P_{(8,10)}$ (2)= 0.01014	0.99998

Setelah didapat peluang perpindahan semut langkah selanjutnya yaitu membangkitkan bilangan acak untuk mendapatkan rute yang akan terpilih.

- Bangkitkan bilangan acak : **0.96987**
- Titik berikutnya yang dipilih : **6**

Perhitungan dilakukan sampai t = 10, sehingga diperoleh 10 lintasan yang selanjutnya dicari lintasan terpendeknya. Lintasan itulah yang selanjutnya dijadikan solusi yang optimal atau penjadwalan yang optimal. Misalkan dari kasus diatas, panjang lintasan yang diperoleh adalah sebagai berikut:

Tabel 3. 20*Tabu list* saat t = 10

M	Lintasan	Panjang Lintasan
1	1-4-3-8-7-10-2-9-6-5	18
2	2-5-1-4-10-6-3-8-9-7	15
3	3-10-2-1-7-5-4-9-8-6	9
4	4-8-6-10-9-5-2-1-3- 7	12
5	5-3-2-10-8-6-1-9-4-7	15
6	6-2-9-4-1-3-8-5-7-10	6
7	7-9-2-6-8-1-3-4-5-10	18
8	8-6-3-5-4-9-2-7-1-10	12
9	9-7-5-4-6-3-2-10-1-8	15
10	10-1-4-2-8 -6-3-5-7-9	15

Dari tabel 3.20 diatas, maka lintasan terpendek yang terpilih adalah lintasan yang dimulai oleh semut ke-6 dengan komposisi lintasan 6 – 2 – 9 – 4 – 1 – 3 – 8 – 5 – 7 – 10 dengan panjang lintasan 6.

3.5 Implementasi Lintasan ke Penjadwalan

Proses mengubah lintasan ke jadwal harus kita kombinasikan dengan jumlah ruangan yang disediakan. Dalam perhitungan manual data telah diasumsikan bahwa ruangan yang disediakan adalah 4 ruangan tiap jamnya dan jam yang tersedia adalah tiga jam. Lintasan yang mempunyai bobot total terkecil adalah 6-2-9-4-1-3-8-5-7-10. Karena asumsi yang digunakan 3 hari maka nantinya kombinasi lintasan digunakan aturan:

1. Titik pertama akan diletakan pada hari ke-1 – Ruang ke-1 – Jam ke-1.
2. Titik berikutnya akan diletakan pada hari ke-2 – Ruang ke-1- jam ke-1.
3. Titik berikutnya akan diletakan pada hari ke-3 – Ruang ke-1- jam ke-1.
4. Titik selanjutnya akan diletakkan kembali pada pada hari ke-1 lagi dengan jam ke-1 tetapi pada ruang ke-2. Begitu pula selanjutnya dengan titik-titik yang lain.

Tabel 3.21 Mengubah lintasan menjadi jadwal hari-1

H1											
J1				J2				J3			
R1	R2	R3	R4	R1	R2	R3	R4	R1	R2	R3	R4
6	10			4				8			

Tabel 3.22 Mengubah lintasan menjadi jadwal hari-2

H2											
J1				J2				J3			
R1	R2	R3	R4	R1	R2	R3	R4	R1	R2	R3	R4
2				1				5			

Tabel 3.23 Mengubah lintasan menjadi jadwal hari-3

H3											
J1				J2				J3			
R1	R2	R3	R4	R1	R2	R3	R4	R1	R2	R3	R4
9				3				7			

Keterangan:
H_i : Hari ke-*i* ;*i* = 1, 2, 3
J_i : Jam ke-*i*;*i* = 1, 2, 3
R_j : Ruang ke-*j* ;*j* = 1, 2, 3, 4
Lintasan : 6-2-9-4-1-3-8-5-7-10 (kode mata kuliah sesuai dengan tabel 3.2).

Jadwal yang dihasilkan dari tabel 3.21, tabel 3.22 dan tabel 3.23 selanjutnya diubah berdasarkan kode atau titik mata kuliah yang telah dibuat pada tabel 3.2.Jadwal yang dihasilkan dapat dituliskan seperti tabel 3.24, tabel 3.25 dan tabel 3.26 :

Tabel 3.24 Jadwal yang dihasilkan hari ke-1

Hari	Jam	Ruang	Kode	Mata kuliah
H1	1	R1	6	Basis Data I (A)
		R2	10	Keamanan Jaringan (A)
	2	R1	4	Algoritma Genetika (B)
	3	R1	8	Rekayasa Perangkat Lunak (A)

Tabel 3.25 Jadwal yang dihasilkan hari ke-2

Hari	Jam	Ruang	Kode	Mata kuliah
H2	1	R1	2	Algoritma Genetika (A)
	2	R1	1	Matematika Diskrit (A)
	3	R1	5	Algoritma Genetika (B)

Tabel 3.26 Jadwal yang dihasilkan hari ke-3

Hari	Jam	Ruang	Kode	Mata kuliah
H2	1	R1	9	Kemanan Jaringan (A)
	2	R1	3	Algoritma Genetika (A)
	3	R1	7	Rekayasa Perangkat Lunak (A)

3.6 Pengosongan *Tabu List*

Apabila belum mencapai iterasi maksimum, maka algoritma perlu di ulang lagi dari langkah 2 dengan *pheromone* yang sudah diperbarui. Disamping itu *tabu list*juga perlu dikosongkan kembali, untuk diisikan lagi dengan urutan titik yang baru pada iterasi berikutnya.

3.7 Perancangan Strukur Data

Struktur data berguna untuk menyimpan data yang akan digunakan pada aplikasi. Pada perancangan ini dibentuk beberapa model struktur data yang digunakan pada aplikasi dengan

menggunakan array record. Bentuk-bentuk struktur data tersebut adalah:

1. Perancangan Struktur Data Graph

Kasus penjadwalan direpresentasikan dalam graph, sehingga terdapat simpul dan garis yang menghubungkan simpul-simpul tersebut.Simpul-simpul dinamakan node atau titik, sedangkan garis yang menghubungkan dinamakan vertek. Struktur data graph dirancang dalam bentuk *text* berbasis *record* yang mempunyai beberapa atribut. Atribut-atribut ini adalah: mata kuliah, waktu, ruang dan pengajar.

2. Perancangan Struktur Titik

Penyimpanan data yang berisi nama node, mata kuliah, dosen, ruang, waktu dan bobot yang dilalui antar node.

3. Perancangan Struktur Data Semut

Struktur data semut dirancang dalam bentuk array dengan atribut-atribut :

a. *Tabu list* yang dimiliki oleh semut tersebut
b. Jalur yang dihasilkan oleh semut tersebut
c. Total bobot setelah semut tersebut berjalan
d. Titik-titik yang telah dikunjungi oleh semut

3.8 Perancangan Ujicoba

Ujicoba dilakukan guna mengetahui kebenaran algoritma semut dalam menyeleseikan masalah penjadwalan mata kuliah. Rancangan ujicoba dibagi menjadi dua tahap, yaitu uji coba rute dengan parameter yang berbeda dan uji coba rute yang dihasilkan pada beberapa kali program yang dijalankan.Di bawah ini merupakan rancangan uji coba jalur dengan parameter yang berbeda yang dihasilkan oleh masing-masing semut pada tiap siklus:

Tabel 3.27 Rancangan ujicobajalur tiap Siklus

Ncmax	τij	α	β	ρ	Lintasan	Panjang lintasan

Nilai α dan β merupakan nilai intensitas dan visibilitas antar titik yang sangat mempengaruhi panjang lintasan. Sedangkan ρ adalah nilai penguapan*pheromone* yang akan mempengaruhi nilai intensitas

jejak *pheromone* dalam setiap siklus. Detail ujicoba adalah sebagai berikut:

1. Nilai α dan β akan diujicoba menggunakan beberapa nilai antara 0 sampai 3. Kemudian akan dilihat bagaimana pengaruhnya terhadap panjang lintasan yang diperoleh dan waktu komputasi yang di butuhkan. Antara nilai α dan β nilainya akan saling dikombinasikan, sehingga bisa diketahui dengan nilai α dan β berapa lintasan terpendek bisa diperoleh.
2. Nilai ρ akan dicoba menggunakan nilai antara 0 sampai 1. Dengan nilai parameter tersebut akandilihat seberapa besar pengaruh penguapan terhadap panjang lintasan yang diperoleh dan pengaruhnya terhadap nilai intensitas *pheromone*.
3. Nilai τ(0) akan dicoba dan dievaluasi menggunakan nilai awal 0.01, yang selanjutnya akan dinaikkan dan diturunkan sesuai dengan hasil lintasan yang diperoleh.
4. Akan dicari kombinasi maksimal dan menghasilkan rute terpendek antara α, β, ρ, τ(0) serta dengan perulangan atau siklus berapa kali akan menghasilkan solusi yang optimal untuk penjadwalan.

Pengujian dilakukan dengan menguji beberapa parameter dan pengaruhnya terhadap waktu komputasi dari perangkat lunak yang akan dibuat. Berikut adalah beberapa parameter yang diuji terhadap waktu komputasi dengan menguji beberapa kali.

Tabel 3.28 Pengaruh α dan β terhadap waktu komputasi

α	B	Banyak Percobaan			Lintasan		Waktu (s)
		P_1	P_2	P_3	Terbaik	Terburuk	

Dimana : P_i : Percobaan ke-*i* (*i* = 1, 2, 3)

Tabel 3.29 Pengaruh ρ terhadap waktu komputasi

ρ	Banyak Percobaan			Lintasan		Waktu (s)
	P_1	P_2	P_3	Terbaik	Terburuk	

Dimana : P_i : Percobaan ke-*i* (*i* = 1, 2, 3)

Tabel 3.30 Pengaruh N terhadap waktu komputasi

N	Banyak Percobaan			Lintasan		Waktu (s)
	P_1	P_2	P_3	Terbaik	Terburuk	

Dimana : P_i : Percobaan ke-*i* (*i* = 1, 2, 3)

3.9 Desain Interface

*Desain interface*atau perancangan antar muka digunakan untuk merancang interaksi antara pengguna dan sistem dengan tujuan mempermudah pengguna dalam menggunakan aplikasi. Dalam aplikasi ini terdapat dua jenis yaitu antarmuka utama dan antarmuka hasil proses.

3.9.1 Antarmuka Utama

Antarmuka utama digunakan sebagai awal dari penggunaan aplikasi penjadwalan mata kuliah menggunakan algoritma koloni semut di jurusan matematika Universitas Brawijaya Malang. Pada bagian ini pengguna bisa memilih data yang ada pada *database* yang akan diproses lebih lanjut. Terdapat beberapa pilihan menu yang disediakan, yaitu:

- *New Document* :Berfungsi untuk membuat data-data penjadwalan yang akan dibuat.
- *Open Document* : Berfungsi untuk membuka file penjadwalan sebelumnya yang selanjutnya bisa dibuat kembali penjadwalan secara otomatis dengan format tertentu.
- *Save* : menyimpan data penjadwalan yang sudah dibuat
- *Export – Import* : Berfungsi untuk mengimport atau mengeksport data jadwal yang terbentuk ke dalam sistem berupa data file bertipe excel.
- *Exit* : keluar dari program aplikasi

3.9.2 Antarmuka Hasil Proses

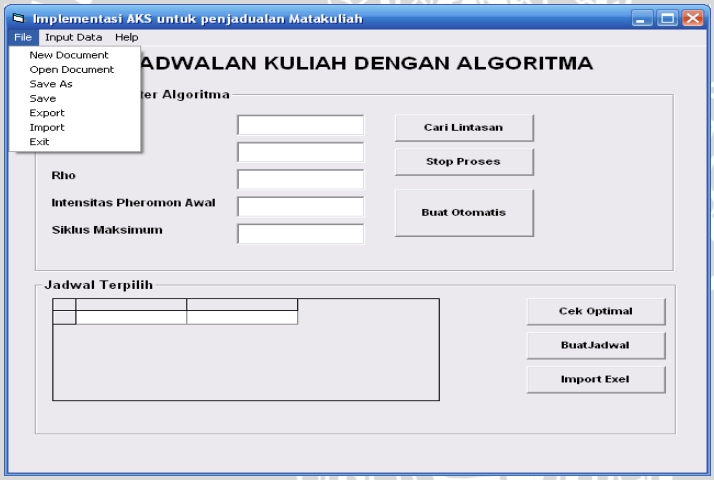
Antarmuka hasil proses, terdapat beberapa pilihan:

- Input manual masing-masing tabel, artinya selain dari file export dari exel, kita bisa melakukan input secara manual

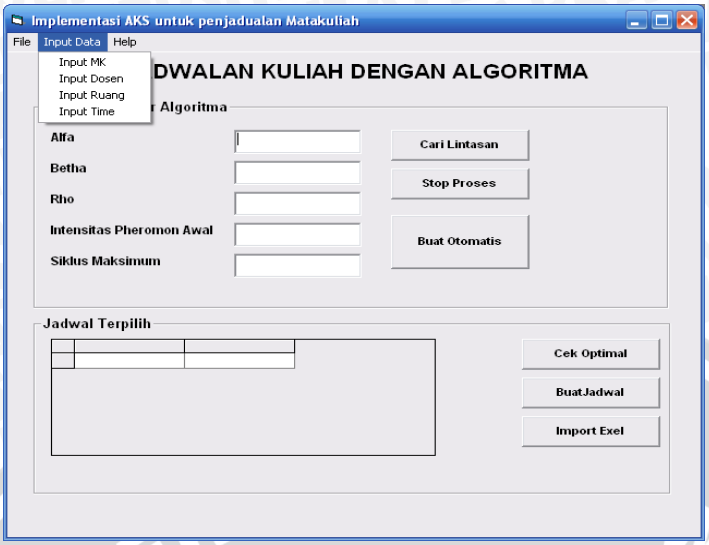
- Proses jadwal, berupa pembuatan jadwal secara otomatis

Berikut gambar rencana implementasi yang akan dibuat, yaitu penjadwalan mata kuliah menggunakan algoritma koloni semut.

Gambar 3.5 Desain *interface1*



Gambar 3.6 Desain *interface 2*



Gambar 3.7 Desain *interface 3*

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Lingkungan Implementasi

Implementasi merupakan representasi rancangan berupa aplikasi penjadwalan matakuliah menggunakan algoritma koloni semut. Adapun yang akan dijelaskan dalam subbab ini meliputi lingkungan perangkat keras dan perangkat lunak.

4.1.1 Lingkungan perangkat keras

Perangkat keras yang digunakan dalam pengembangan dan pengujian aplikasi penjadwalan Mata kuliah ini adalah sebuah *Notebook* dengan spesifikasi sebagai berikut :

1. Prosesor Intel(R) Core(TM) i3 CPU M 330 @2.13GHz
2. Memori 2 Gb
3. Harddisk 320 Gb
4. Monitor 13”
5. Keyboard
6. Mouse

4.1.2 Lingkungan perangkat lunak

Perangkat lunak yang digunakan dalam pengembangan program enkripsi citra digital dan uji coba adalah:

1. Sistem operasi Windows 7 64-bit sebagai tempat aplikasi dijalankan.
2. Visual Studio2010 c#sebagai *programming software development* dalam pembuatan aplikasi Penjadwalan Mata kuliah.

4.2 Implementasi Program

Berdasarkan analisa dan perancangan proses yang telah dipaparkan pada Bab III, maka pada bab ini akan dijelaskan proses implementasinya.

4.2.1 Implementasi *Load* Mata Kuliah

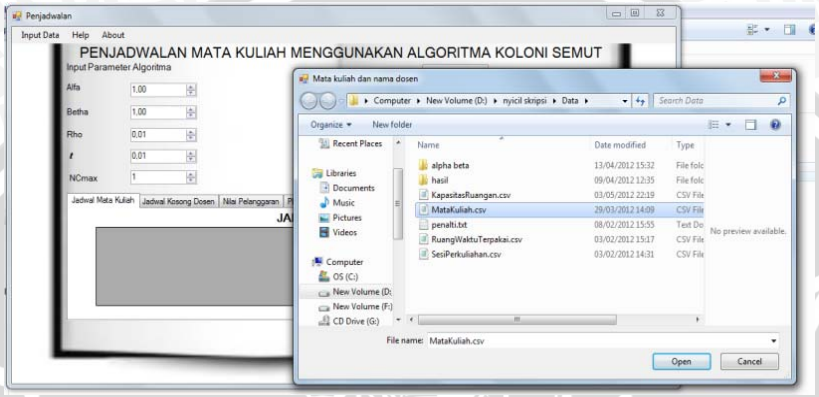
Tahap awal yang dilakukan dalam penjadwalan mata kuliah adalah dengan *load* mata kuliah yang tersedia untuk dijadwalkan yang bertipe csv. Dimana terdapat informasi berupa program studi, mata kuliah, kelas, sks, dosen pengajar.*Sourcode* untuk load mata kuliah dapat dilihat pada *sourcode* 4.1

```
publicstaticIEnumerable<Titik> Load(string
lokasiFile)
        {
            Daftar.Clear();
var indeks = 0;
foreach (var baris
inFile.ReadAllLines(lokasiFile))
            {
//membaca data dengan pemisahnya (;)
var pisahan = baris.Split(new[] { ';' },
StringSplitOptions.RemoveEmptyEntries);
var sks = int.Parse(pisahan[1]);
//pemecahan sks 3 menjadi 2 dan 1
if (sks < 3)
                {
var t = newTitik() { NamaMatakuliahLengkap =
pisahan[0], SKS = int.Parse(pisahan[1]), Kelas
= pisahan[2], Semester = int.Parse(pisahan[3]),
NamaDosen = pisahan[4], Indeks = indeks++ };
                    Daftar.Add(t);
yieldreturn t;
                }
else
                {
var t = newTitik() { NamaMatakuliahLengkap =
pisahan[0], SKS = 2, Kelas = pisahan[2],
Semester = int.Parse(pisahan[3]), NamaDosen =
pisahan[4], Indeks = indeks++ };
var tx = newTitik() { NamaMatakuliahLengkap =
pisahan[0], SKS = int.Parse(pisahan[1]) - 2,
Kelas = pisahan[2], Semester =
int.Parse(pisahan[3]), NamaDosen = pisahan[4],
Indeks = indeks++ };
                    Daftar.Add(t);
yieldreturn t;
                    Daftar.Add(tx);
yieldreturn tx;
```

```
                }
            }
        }
```

Sourcecode 4.1 Load Mata Kuliah

Untuk lebih jelasnya dapat kita lihat implementasi dari *sourcecode* load mata kuliah seperti pada gambar 4.1. Untuk memulai proses pembuatan jadwal terlebih dahulu masukkan data mata kuliah dengan cara input data pilih file dengan nama matakuliah. Dapat dilihat pada gambar 4.1.



Gambar 4.1implementasi load mata kuliah

4.2.2 Implementasi *Load* Jam Perkuliahan

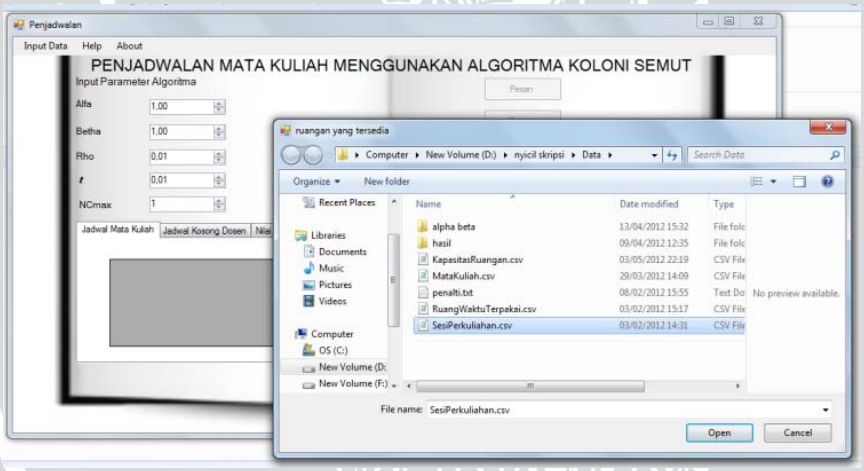
Tahap selanjutnya yang dilakukan dalam penjadwalan mata kuliah dengan *load* jam perkuliahan yang akan diselenggarakan terdapat alokasi waktu yang diberikan tiap-tiap SKS. *Sourcecode* untuk load mata kuliah dapat dilihat pada *sourcecode* 4.2

```
publicstaticIEnumerable<SlotWaktu> LoadWaktu(string
fileLokasi)
        {
            ArtiWaktu.Clear();
            Daftar.Clear();
var file = File.ReadAllLines(fileLokasi);
foreach (var item in file)
            {
```

```
var pisahan = item.Split(new[] { ';' },
StringSplitOptions.RemoveEmptyEntries);
var jamKe = int.Parse(pisahan[2]);
var awal = pisahan[0];
var akhir = pisahan[1];
                ArtiWaktu.Add(jamKe, newTuple<string,
string>(awal, akhir));
for (int hari = 1; hari <= 5; hari++)
                {
if (jamKe==5&&hari ==5)
                    {
continue;
                    }
var hasil = newSlotWaktu() { Hari = hari, JamKe =
jamKe };
                    Daftar.Add(hasil);
yieldreturn hasil;
                }
            }
        }
```

Sourcecode 4.2*Load* Jam Perkuliahan

Untuk mengetahui implementasi dari *sourcecodeload* jam perkuliahan dapat dilihat padagambar 4.2. Proses selanjutnya untuk pembuatan jadwal masukkan data mata kuliah dengan cara input data pilih file dengan SesiPerkuliahan.



Gambar 4.2implementasi load jam perkuliahan

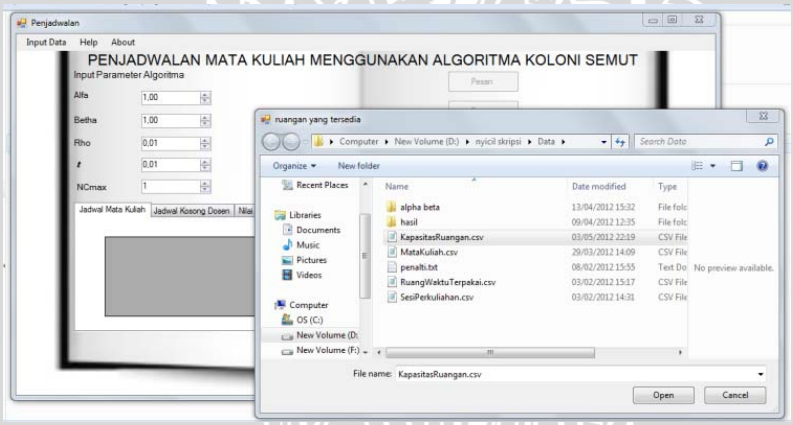
4.2.3 Implementasi *Load* Ruangan

Tahap ini meload file ruangan yang bertipe csv yang berisi ruangan yang dapat digunakan dalam proses perkuliahan dimana terdapat nama ruangan serta kapasitas ruangan. Berikut ini *sourcecode* untuk Load ruangan dapat dilihat pada *sourcecode*4.3 :

```
publicstaticIEnumerable<Ruang> LoadRuangan(String
fileLokasi)
        {
            Daftar.Clear();
var file = File.ReadAllLines(fileLokasi);
foreach (var item in file)
            {
var pisahan = item.Split(new[] { ';' },
StringSplitOptions.RemoveEmptyEntries);
var NamaRuangan = pisahan[0];
var DayaTampung = int.Parse(pisahan[1]);
var hasil = newRuang() { Nama = NamaRuangan,
DayaTampung = DayaTampung };
                Daftar.Add(hasil);
yieldreturn hasil;
            }
        }
```

Sourcecode 4.3 Load Ruangan

Untuk mengetahui implementasi dari *sourcecodeload*ruangan perkuliahan dapat dilihat pada gambar 4.3. Proses selanjutnya untuk pembuatan jadwal masukkan data ruangan dengan cara input data pilih file dengan KapasitasRuangan.



Gambar 4.3implementasi load ruangan

4.2.4 Implementasi *Load* Ruang Waktu Terpakai

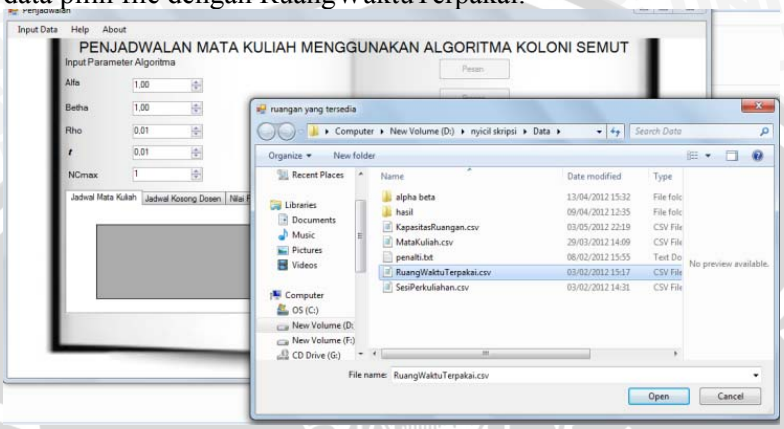
Load ruang dan waktu yang bertipe csv digunakan untuk mengetahui ruang kuliah dan waktu yang dipakai oleh jurusan lain sehingga dapat menghindari jadwal yang bentrok dengan jurusan lain. *Sourcecode* untuk load ruang dan waktu terpakai dapat dilihat pada *sourcecode* 4.4.

```
publicstaticIEnumerable<RuangWaktuTersedia>
Load(string lokasiTerpakai)
        {
var daftarTerpakai =
File.ReadAllLines(lokasiTerpakai);

            Daftar = newList<RuangWaktuTersedia>();

foreach (var waktu inSlotWaktu.Daftar)
            {
foreach (var ruang inRuang.Daftar)
                {
                    Daftar.Add(newRuangWaktuTersedia()
{ Ruangan = ruang, Waktu = waktu });
                }
            }


foreach (var item in daftarTerpakai)
            {
var pisahan = item.Split(new[] { ';' },
StringSplitOptions.RemoveEmptyEntries);
var slotTerpakai = SlotWaktu.Jabarkan(pisahan[1],
pisahan[2], pisahan[3]);
foreach (var slot in slotTerpakai)
                {
                    Daftar.RemoveAll(x =>
x.Ruangan.Nama == pisahan[0] && x.Waktu.Hari ==
slot.Hari && x.Waktu.JamKe == slot.JamKe);
                }
            }
            Daftar = Daftar.OrderBy(x =>
x.Waktu).OrderBy(x => x.Ruangan.Nama).ToList();
foreach (var item in Daftar)
            {
```

```
yieldreturn item;
            }

        }
```

Sourcecode 4. 4 Ruang Waktu Terpakai

Untuk mengetahui implementasi dari *sourcecodeload* ruang waktu terpakai dapat dilihat pada gambar 4.4. Proses selanjutnya untuk pembuatan jadwal masukkan data ruangan dengan cara input data pilih file dengan RuangWaktuTerpakai.



Gambar 4.4implementasi load ruang waktu terpakai

4.2.5 Implementasi Pemecahan Mata Kuliah

Setelah didapatkan semua mata kuliah yang akan ditawarkan maka terlebih dahulu dilakukan pemecahan SKS. Pemecahan SKS dilakukan sesuai dengan aturan yang diterapkan oleh fakultas MIPA.

4.2.5.1 Pemecahan SKS Menurut MIPA

SKS yang berjumlah 3 SKS akan dipecah menjadi dua pertemuan yaitu 2 SKS dan 1 SKS, untuk 4 SKS dibuat 2 sks dan 2 sks. *Sourcecode* 4.5menunjukkan pemecahan SKS :

```
publicstaticIEnumerable<Titik> Load(string
lokasiFile)
```

```
        {
            Daftar.Clear();
var indeks = 0;
foreach (var baris inFile.ReadAllLines(lokasiFile))
            {
//membaca data dengan pemisahnya (;)
var pisahan = baris.Split(new[] { ';' },
StringSplitOptions.RemoveEmptyEntries);
var sks = int.Parse(pisahan[1]);
//pemecahan sks 3 menjadi 2 dan 1
if (sks < 3)
                {
var t = newTitik() { NamaMatakuliahLengkap =
pisahan[0], SKS = int.Parse(pisahan[1]), Kelas =
pisahan[2], Semester = int.Parse(pisahan[3]),
NamaDosen = pisahan[4], Indeks = indeks++ };
                    Daftar.Add(t);
yieldreturn t;
                }
else
                {
var t = newTitik() { NamaMatakuliahLengkap =
pisahan[0], SKS = 2, Kelas = pisahan[2], Semester =
int.Parse(pisahan[3]), NamaDosen = pisahan[4],
Indeks = indeks++ };
var tx = newTitik() { NamaMatakuliahLengkap =
pisahan[0], SKS = int.Parse(pisahan[1]) - 2, Kelas =
pisahan[2], Semester = int.Parse(pisahan[3]),
NamaDosen = pisahan[4], Indeks = indeks++ };
                    Daftar.Add(t);
yieldreturn t;
                    Daftar.Add(tx);
yieldreturn tx;

                }
            }
        }
```

Sourcecode 4.5 Pemecahan SKS Menurut MIPA

4.2.5.2 Pemecahan SKS Sesuai Sistem

Pecahan mata kuliah setelah menurut aturan mipa dipecah lagi sesuai dengan kebutuhan sistem dimana semua mata kuliah dipecah menjadi 1 SKS. Pemecahan ini bertujuan untuk menyesuaikan dengan jam perkuliahan yang telah disusun setiap SKS memiliki waktu 50 menit. Berikut sourcode 4.6 dari pemecahan menjadi 1 SKS.

```
publicstaticvoid TambahKosong(int jumlahTotal)
        {
// memuat mata kuliah berdasarkan sks
var jumlahSekarang = Daftar.Sum(x => x.SKS);
var idx = Daftar.Count;
// memuat sisa mata kuliah
            Daftar.AddRange(Enumerable.Range(0,
jumlahTotal - jumlahSekarang).Select(x =>newTitik()
{ SKS = 1,Indeks = x+idx }));
        }
```

Sourcecode 4.6 Pemecahan SKS Sesuai Sistem

4.2.6 Implementasi Pemberian Aturan

Pemberian aturandimaksudkan untuk mengatur jadwal lebih tertata dan lebih terstruktur sehingga tidak akan terjadi bentrok. Serta memberikan nilai dari pelanggaran apabila pelanggaran itu terjadi. Berikut sourcode 4.7 dari pemberian aturan.

```
publicdouble Jarak(Titik tLain)
        {
if
(String.IsNullOrEmpty(tLain.NamaMatakuliahLengkap)
|| String.IsNullOrEmpty(this.NamaMatakuliahLengkap))
            {
return 0.0;
            }
var nilai = 0.0;


if (this.Semester == tLain.Semester &&this.Kelas ==
tLain.Kelas &&this.Prodi == tLain.Prodi)
            {
                nilai = 0.25;
            }
if (this.NamaDosen == tLain.NamaDosen)
            {
                nilai = 0.33;
            }
if (this.Semester == tLain.Semester &&this.Kelas ==
tLain.Kelas &&this.Prodi == tLain.Prodi
&&this.NamaDosen == tLain.NamaDosen)
            {
                nilai = 0.66;
            }

return nilai;
        }
```

Sourcecode 4.7 Pemberian Aturan

4.2.7 Implementasi Update Pheronome

Implementasi update pheronome digunakan untuk perbaikan *pheronome* agar mendapatkan nilai yang terbaik dimana hasil yang tersebut akan digunakan untuk tiap generasi.

4.2.7.1 Pemberian Nilai *pheronome* awal

Pemberian nilai penalti awal yang dimaksud adalah memberikan nilai default awal dari pheronome semut yaitu 0,01. Pemberian nilai awal ini agar kecenderungan semut berjalan sama dari titik ke titik selanjutnya. Berikut sourcode 4.8 dari pemberian nilai penalti awal

```
publicstaticdouble[,]
```

```
matrixJejakFeromon(thisIEnumerable<Titik> daftar,
double[,] matrixVisibilitas)
        {

var matrix = newdouble[daftar.Count(), daftar.Count()];
for (int i = 0; i < daftar.Count(); i++)
            {
for (int j   = 0; j < daftar.Count(); j++)
                {
// inisialisasi awal feromon
                    matrix[i,j] = Generasi.FAwal;
                }
            }


Semut.matrixFeromon = matrix;
Semut.matrixVisibilitas = matrixVisibilitas;
Semut.DR = daftar.ToList();
return matrix;
        }
```

Sourcecode 4.8 Pemberian Nilai *Pheronome* awal

4.2.7.2 Pemberian *update*Nilai *Pheronome*

Pemberian nilai update yang dimaksud yaitu perbaikan nilai setelah generasi pertama. Nilai yang dimasukkan berdasarkan nilai yang telah ditentukan. Ini bertujuan agar mendapatkan jadwal yang paling baik dengan nilai pelanggaran yang seminimal mungkin. Berikut sourcecode 4.9 pemberian u*pdate* nilai *pheronome* :

```
publicstaticTuple<Semut, Semut> GenerasiBerikutnya()
        {
var dpx = Enumerable.Range(0,
DR.Count()).AsParallel().WithExecutionMode(ParallelExecu
tionMode.ForceParallelism).Select(x =>newSemut(DR, x,
Generasi.Pertama)).OrderBy(x => x.Nilai).ToList();
Generasi.Pertama = false;
            ParaTerbaik = dpx.Take(10).ToList();

var semutTerbaik = ParaTerbaik.First();
var terburuk = dpx.Last();
//202
//var semutTerbaik = Enumerable.Range(0,
DR.Count()).Select(x => new
```

```
Semut(DR)).OrderByDescending(x => x.Nilai).First();

            _DaftarPasanganDilewati =
newHashSet<Tuple<int, int>>();
for (int index = 0; index < DR.Count - 1; index++)
            {
var a = semutTerbaik.TelahDilewati[index].Indeks;
var b = semutTerbaik.TelahDilewati[index + 1].Indeks;
//rumus update pheronome = 1-(rho.pheronome awal)+
rho.1/jarak terpendek
                matrixFeromon[a, b] = 1 - (Generasi.Rho
* matrixFeromon[a, b]) + Generasi.Rho /
semutTerbaik.Nilai;
                matrixFeromon[b, a] = 1 - (Generasi.Rho
* matrixFeromon[b, a]) + Generasi.Rho /
semutTerbaik.Nilai;

_DaftarPasanganDilewati.Add(newTuple<int, int>(a, b));
            }
for (int a = 0; a < matrixFeromon.GetLength(0); a++)
            {
for (int b = 0; b < a; b++)
                {
if (!_DaftarPasanganDilewati.Contains(newTuple<int,
int>(a, b)))
                    {
                        matrixFeromon[a, b] =
Math.Max(0, matrixFeromon[a, b] - Generasi.Rho);
                        matrixFeromon[b, a] =
Math.Max(0, matrixFeromon[b, a] - Generasi.Rho);
                    }
                }
            }
Generasi.SejarahFeronom.Add(matrixFeromon.Select(x =>
x));
returnnewTuple<Semut, Semut>(semutTerbaik, terburuk);
        }
```

Sourcecode 4.9 Pemberian Nilai *update* nilai *Pheronome*

4.2.8 Implementasi Penggabungan SKS

Proses penggabungan mata kuliah ini dilakukan karena pada proses sebelumnya SKS dipecah menjadi 1 SKS semua. Pada proses ini SKS dikembalikan lagi sesuai dengan aturan yang diterapkan oleh

MIPA dan ditampilkan pada jadwal yang telah terpilih. *Sourcecode*4.10implementasi penggabungan SKS.

```
publicstaticDictionary<Titik, JadwalGabung>
DaftarGabung(Dictionary<Titik, RuangWaktuTersedia>
Jadwal)
        {

var hasil = newDictionary<Titik, JadwalGabung>();
var sebelum = newKeyValuePair<Titik,
RuangWaktuTersedia>(newTitik(),
newRuangWaktuTersedia());

var tes = Jadwal.OrderBy(x => x.Value.Waktu).GroupBy(x
=> x.Value.Ruangan).SelectMany(x => x);
foreach (var item in Jadwal.GroupBy(x => x.Key.Indeks))
            {
if (item.Count() > 1)
                {
var terurut = item.OrderBy(x => x.Value.Waktu).ToList();

                    hasil.Add(terurut[0].Key,
newJadwalGabung()
                    {
                        awal = terurut[0].Value.Waktu,
                        Ruang =
terurut[0].Value.Ruangan.ToString(),
                        Waktu =
SlotWaktu.ArtiHari[terurut[0].Value.Waktu.Hari] + " " +
SlotWaktu.ArtiWaktu[terurut[0].Value.Waktu.JamKe].Item1
+ "-" +
SlotWaktu.ArtiWaktu[terurut[1].Value.Waktu.JamKe].Item2
                    });
                }
else
                {
var terurut = item.ToList();

                    hasil.Add(terurut[0].Key,
newJadwalGabung()
                    {
                        awal = terurut[0].Value.Waktu,
                        Ruang =
terurut[0].Value.Ruangan.ToString(),
                        Waktu =
SlotWaktu.ArtiHari[terurut[0].Value.Waktu.Hari] + " " +
SlotWaktu.ArtiWaktu[terurut[0].Value.Waktu.JamKe].Item1
```

```
+ "-" +
SlotWaktu.ArtiWaktu[terurut[0].Value.Waktu.JamKe].Item2
                    });
                }
            }
return hasil.OrderBy(x => x.Value.awal).ToDictionary(x
=> x.Key, x => x.Value);

        }
```

Sourcecode 4.10 Penggabungan SKS

4.2.9 Implementasi Pemesanan Jadwal

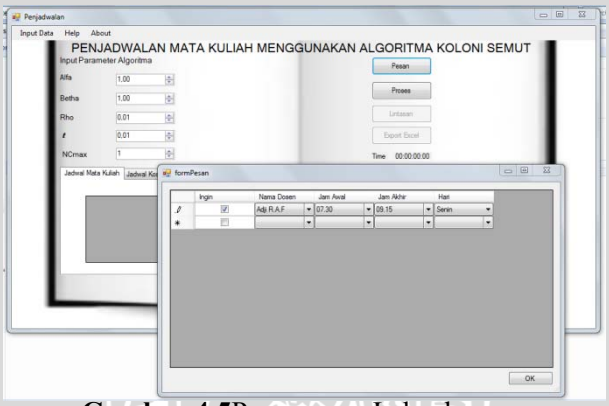
Proses pemesanan mata kuliah ini dilakukan apabila terdapat dosen yang menginginkan waktu tertentu untuk mengajar. Dosen bisa menentukan jam perkuliahan dan hari yang diinginkan.*Sourcecode* 4.11 Implementasi Pemesanan Jadwal.

```
privatevoid button1_Click(object sender, EventArgs e)
        {
Pesanan.Daftar.Clear();
for (int index = 0; index < dataGridView1.Rows.Count-1;
index++)
            {
if (dataGridView1.Rows[index].Cells[0].Value == null)
                {

dataGridView1.Rows[index].Cells[0].Value = false;
                }
if (dataGridView1.Rows[index].Cells[0].Value == "true")
                {

dataGridView1.Rows[index].Cells[0].Value = true;
                }
var SlotAwal = newSlotWaktu() { JamKe =
(int)dataGridView1.Rows[index].Cells[2].Value, Hari =
(int)dataGridView1.Rows[index].Cells[4].Value };
var SlotAkhir = newSlotWaktu() { JamKe =
(int)dataGridView1.Rows[index].Cells[3].Value, Hari =
(int)dataGridView1.Rows[index].Cells[4].Value };
if (SlotAwal>=SlotAkhir)
                {
var sem = SlotAkhir;
```

```
                    SlotAkhir = SlotAwal;
                    SlotAwal = sem;
                }
Pesanan.Daftar.Add(newPesanan()
                    {
                        Ingin =
(bool)dataGridView1.Rows[index].Cells[0].Value,
                        NamaDosen =
(string)dataGridView1.Rows[index].Cells[1].Value,
                        SlotAwal = SlotAwal,
                        SlotAkhir = SlotAkhir
                    });

            }
Pesanan.Save();
this.Close();
        }
```

Sourcecode 4.11 Pemesana Jadwal

Untuk mengetahui implementasi dari *sourcecode* pemesanan mata kuliah dapat dilhat pada gambar 4.5.



Gambar 4.5 Pemesanan Jadwal

Pada gambar 4.5 merupakan proses pemesanan jadwal. Pemesanan jadwal dilakukan apabila salah satu atau beberapa dosen menginginkan waktu-waktu tertentu untuk mengajar. Langkah yang dilakuakan adalah dengan memasukkan nama dosen jam awal yang

diinginkan, jam akhir, hari dan beri ceklist lalu tekan oke. Untuk melakukan proses maka klik button proses dapat dilihat di gambar 4.3.

4.2.10 Implementasi Lintasan

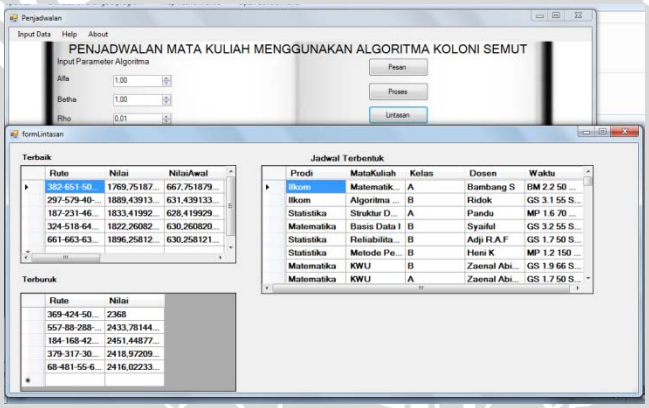
Lintasan merupakan daftar rute yang dilewati oleh semut. Semut bergerak menuju titik selanjutnya berdasarkan *pheronome* yang paling kuat. Rute inilah yang akan diterjemahkan menjadi jadwal.*Sourcecode* 4.12 Implementasi Lintasan

```
publicstaticTuple<Semut, Semut> GenerasiBerikutnya()
        {
var dpx = Enumerable.Range(0,
DR.Count()).AsParallel().WithExecutionMode(ParallelExecu
tionMode.ForceParallelism).Select(x =>newSemut(DR, x,
Generasi.Pertama)).OrderBy(x => x.Nilai).ToList();
Generasi.Pertama = false;
            ParaTerbaik = dpx.Take(10).ToList();

var semutTerbaik = ParaTerbaik.First();
var terburuk = dpx.Last();
//202
//var semutTerbaik = Enumerable.Range(0,
DR.Count()).Select(x => new
Semut(DR)).OrderByDescending(x => x.Nilai).First();

            _DaftarPasanganDilewati =
newHashSet<Tuple<int, int>>();
for (int index = 0; index < DR.Count - 1; index++)
            {
var a = semutTerbaik.TelahDilewati[index].Indeks;
var b = semutTerbaik.TelahDilewati[index + 1].Indeks;
//rumus update pheronome = 1-(rho.pheronome awal)+
rho.1/jarak terpendek
                matrixFeromon[a, b] = 1 - (Generasi.Rho
* matrixFeromon[a, b]) + Generasi.Rho /
semutTerbaik.Nilai;
                matrixFeromon[b, a] = 1 - (Generasi.Rho
* matrixFeromon[b, a]) + Generasi.Rho /
semutTerbaik.Nilai;

_DaftarPasanganDilewati.Add(newTuple<int, int>(a, b));
            }
for (int a = 0; a < matrixFeromon.GetLength(0); a++)
            {
```

```
for (int b = 0; b < a; b++)
                {
if (!_DaftarPasanganDilewati.Contains(newTuple<int,
int>(a, b)))
                    {
                        matrixFeromon[a, b] =
Math.Max(0, matrixFeromon[a, b] - Generasi.Rho);
                        matrixFeromon[b, a] =
Math.Max(0, matrixFeromon[b, a] - Generasi.Rho);
                    }
                }
            }
Generasi.SejarahFeronom.Add(matrixFeromon.Select(x =>
x));
returnnewTuple<Semut, Semut>(semutTerbaik, terburuk);
        }
```

Sourcecode 4.12 Implementasi Lintasan

Untuk mengetahui implementasi dari sourcecode implementasi lintasan dapat dilihat pada gambar 4.6.



Gambar 4.6 Implementasi Lintasan

4.2.11 Implementasi Nilai Pelanggaran

Nilai pelanggaran diberikan apabila terdapat beberapa nilai penalti yang dilanggar dimana pemberian nilai penalti. Beberapa

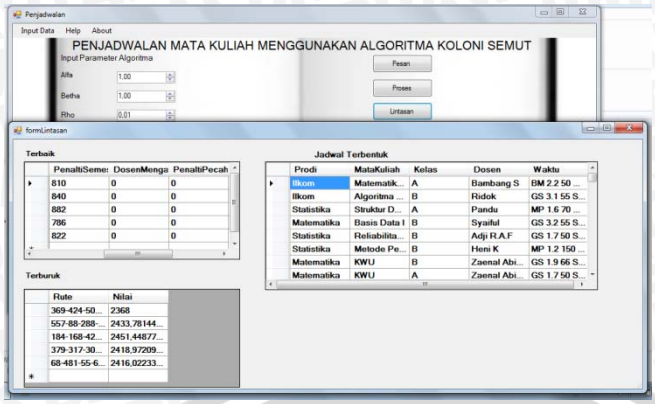
nilai penalti diantaranya yaitu semester awal di jadwalkan dengan jam yang lebih pagi, pecahan SKS mata kuliah tidak boleh dijadwalkan di hari yang sama, area mengajar untuk dosen yang mengajar berurutan tempatnya tidak berjauhan.*Sourcecode* 4.13implementasi nilai pelanggaran.

```
if (stabil)
                {
foreach (var pesanan inPesanan.Daftar.Where(x =>
x.Ingin).GroupBy(x => x.NamaDosen).ToList())
                    {
foreach (var jadwal in Jadwal.Where(x =>
!sudahPindah.Contains(x.Key) && x.Key.NamaDosen ==
pesanan.Key && !sudahPindah.Contains(x.Key)).ToList())
                        {
if (jadwal.Key.pasangan != 0)
                            {
continue;
                            }
if (waktuKosong.Any(x => pesanan.Any(y => x.Waktu >=
y.SlotAwal && x.Waktu <= y.SlotAkhir && !Jadwal.Any(j =>
j.Key.NamaDosen == jadwal.Key.NamaDosen && j.Value.Waktu
== x.Waktu))))
                            {
var pengganti = waktuKosong.First(x => pesanan.Any(y =>
x.Waktu >= y.SlotAwal && x.Waktu <= y.SlotAkhir &&
!Jadwal.Any(j => j.Key.NamaDosen == jadwal.Key.NamaDosen
&& j.Value.Waktu == x.Waktu)));
                                stabil = false;
                                waktuBebas =
Jadwal[jadwal.Key];
                                Jadwal[jadwal.Key] =
pengganti;

waktuKosong.Remove(pengganti);

waktuKosong.Add(waktuBebas);

sudahPindah.Add(jadwal.Key);


                            }
                        }
```

```
                    }
///sudahPindah.Clear();
                }

            }
while (!stabil);

            NilaiPenaltiSemesterAwal = Jadwal.Count(x =>
x.Key.Semester < 5 && x.Value.Waktu.JamKe > 4) *
pinaltiSemesterAwal;
var totalPinalti = NilaiPenaltiSemesterAwal;

var jadwalBerdasarMK = Jadwal.GroupBy(x =>
x.Key.NamaMatakuliahLengkap + x.Key.NamaDosen +
x.Key.Kelas);

// pecah sks dan loncati 1 hari


            NilaiPenaltiHariBersebelahan =
jadwalBerdasarMK.Count(grup => grup.Select(x =>
x.Value.Waktu.Hari).Bersebelahan()) * pinaltiPecahanSKS;
            totalPinalti +=
NilaiPenaltiHariBersebelahan;
// area mengajar jauh dan dekat
var jadwalBerdasarDosen = Jadwal.GroupBy(x =>
x.Key.NamaDosen);
            NilaiPenaltiArea =
jadwalBerdasarDosen.Sum(grup => grup.Select(g =>
g.Value).JumlahBerjauhan()) * pinaltiArea;
            totalPinalti += NilaiPenaltiArea;


            nilaiAwal += totalPinalti;
```

Sourcecode 4.13 Implementasi Nilai Pelanggaran

Untuk mengetahui implementasi dari sourcecode implementasi pelanggaran dapat dilihat pada gambar 4.7. Terdapat informasi pelanggaran dari masing-maing nilai penalti yang dihasilkan.



Gambar 4.7 Implementasi Nilai Pelanggaran

4.2.12 Implementasi Jadwal Kosong Dosen

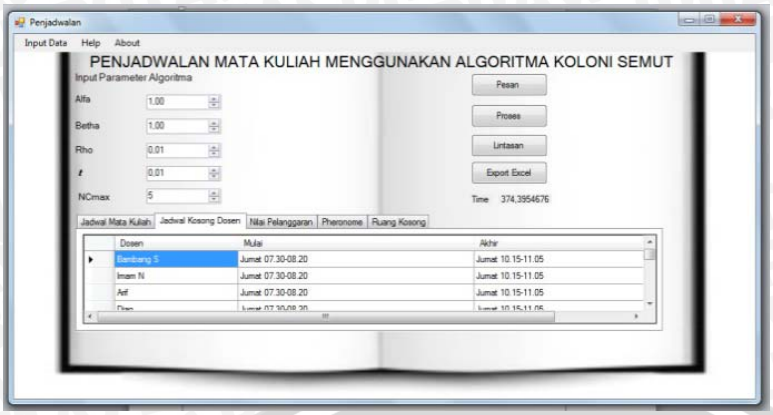
Penjadwalan mata kuliah juga dapat memberikan informasi dosen memiliki jadwal kosong. Jadwal kosong menampilkan jam awal, jam akhir serta hari dosen tidak mengajar. Berikiut *Sourcecode* 4.14 implementasi jadwal kosong dosen.

```
publicstaticIEnumerable<Tuple<string, SlotWaktu,
SlotWaktu>> DaftarKosong(Dictionary<Titik,
RuangWaktuTersedia> Jadwal)
        {
SlotWaktu awal, sebelum;
foreach (var Dosen inTitik.Daftar.Select(x =>
x.NamaDosen).Distinct().Where(x =>
!String.IsNullOrEmpty(x)))
            {
                awal = sebelum = null;
foreach (var waktu inSlotWaktu.Daftar.Where(x =>
!Jadwal.Any(j => j.Key.NamaDosen == Dosen &&
j.Value.Waktu == x)).OrderBy(x => x))
                {

if (awal == null)
                    {
                        awal = waktu;
continue;
                    }
```

```
if (sebelum == null)
                    {
if (waktu.Hari == awal.Hari && waktu.JamKe - awal.JamKe
== 1)
                        {
                            sebelum = waktu;

                        }
continue;
                    }
if (waktu.Hari == sebelum.Hari && waktu.JamKe -
sebelum.JamKe == 1)
                    {
                        sebelum = waktu;
continue;
                    }
yieldreturnnewTuple<string, SlotWaktu, SlotWaktu>(Dosen,
awal, sebelum);
                    awal = waktu;
                    sebelum = waktu;
                }
            }
        }
```

Sourcecode 4.14 Implementasi Jadwal Kosong Dosen

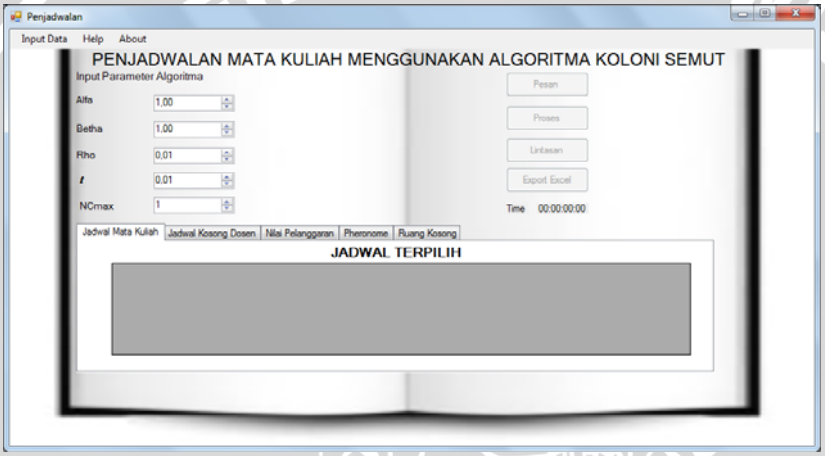
Untuk mengetahui implementasi dari sourcecode implementasi jadwal kosong dosendapat dilihat pada gambar 4.8. Terdapat informasi pelanggaran dari masing-maing nilai penalti yang dihasilkan.



Gambar 4.8 Implementasi Jadwal Kosong Dosen

4.3 Implemantasi Antar Muka

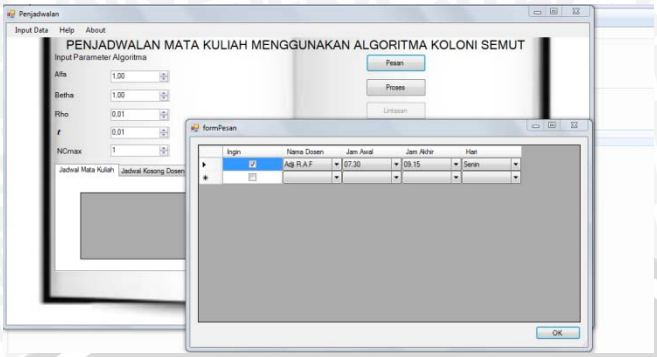
Berdasarkan rancangan antar muka yang telah dikemukakan pada bab 3 maka dihasilkan antar muka pada gambar 4.1



Gambar 4.9 Antarmuka utama

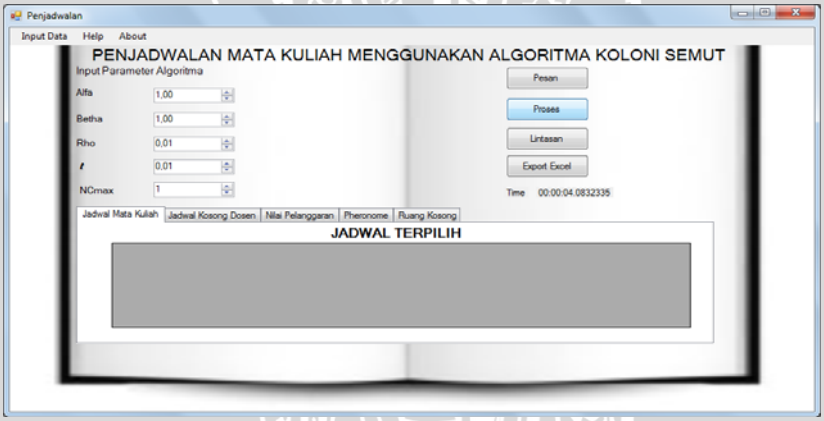
Untuk melakukan proses pembuatan jadwal mata kuliah maka user harus terlebih dahulu memasukkan data. Data yang dimasukkan yaitu data matakuliah, jam, ruang perkuliahan dan ruang waktu yang terpakai yang bertipe csv. Setelah memasukkan data maka terlebih

dahulu set input parameter algoritma alpha, betha, rho, serta jumlah generasinya. Selanjutnya untuk melakukan proses pemesanan dapat dilihat pada gambar 4.9



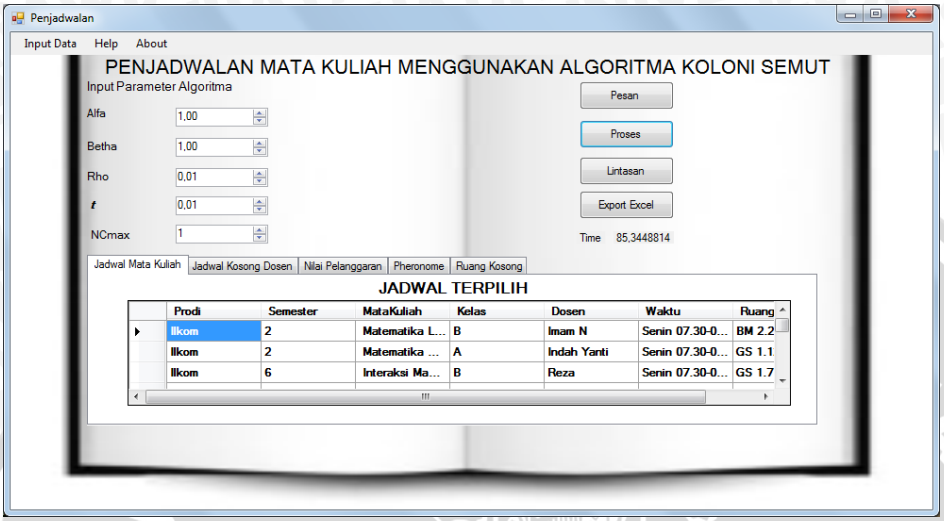
Gambar 4.10 Pemesanan Jadwal

Pada gambar 4.10 merupakan proses pemesanan jadwal. Pemesanan jadwal dilakukan apabila salah satu atau beberapa dosen menginginkan waktu-waktu tertentu untuk mengajar. Langkah yang dilakukan adalah dengan memasukkan nama dosen jam awal yang diinginkan, jam akhir, hari dan beri ceklist lalu tekan oke. Untuk melakukan proses maka klik button proses dapat dilihat di gambar 4.11.



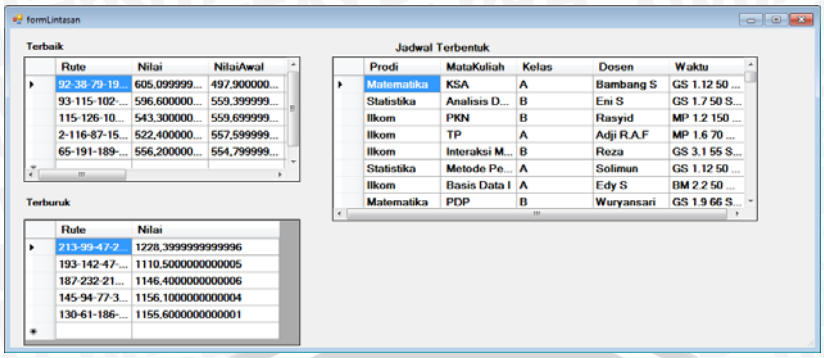
Gambar 4.11 Proses Pembuatan jadwal

Setelah memasukkan data, input parameter algoritma serta pemesanan jadwal maka untuk menghasilkan jadwal tekan proses. Pada saat tekan proses program berjalan seperti gambar 4.3 terdapat waktu yang berjalan dimana waktu yang yang berjalan dan akan berhenti apabila proses pembuatan jadwal selesai dilakukan. Dapat dilihat pada gambar 4.12 hasil dari pembuatan jadwal.



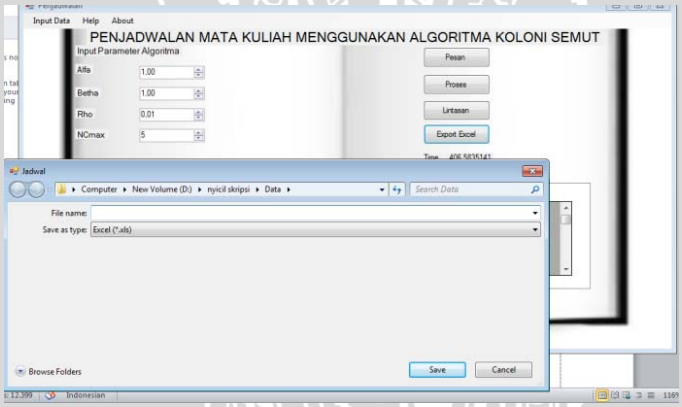
Gambar 4.12 Hasil pembuatan jadwal

Pada gambar 4.12 merupakan hasil dari pembuatan jadwal terdapat informasi jadwal mata kuliah yang terpilih, jadwal kosong dosen, nilai pelanggaran serta pheronome dari masing-masing generasi, serta mengetahui waktu yang dibutuhkan selama program itu berjalan. Untuk mengetahui lintasan yang yang terbentuk dari masing-masing generasi dapat dilihat dengan klik button lintasan seperti pada gambar 4.13.



Gambar 4.13 Lintasan yang terbentuk

Pada gambar 4.13 menunjukkan rute terbaik dari masing-masing generasi, rute terburuk serta jadwal yang dihasilkan oleh rute tersebut. Pada masing-masing rute menunjukkan nilai pelanggaran yang terbuat dari nilai awal, nilai penalti semester awal, nilai penalti, dosen mengajar di waktu yang sama, pecahan SKS dalam satu hari, penalti bersebelahan, penalti area. Untuk menutup lintasan terbentuk tekan tanda silang merah di atas pojoik kanan. Untuk memudahkan dalam melihat informasi jadwal yang telah terbentuk *exsport*ke dalam file excel dengan klik button export excel seperti pada gambar 4.14.



Gambar 4.14 Export excel

Pada gambar 4.14 menunjukkan export file ke dalam excel untuk mempermudah dalam mencerna informasi yang dihasilkan oleh program. Pilih lokasi file dan berikan nama file untuk yang pertama merupakan jadwal mata kuliah dan yang kedua merupakan jadwal kosong dosen dan tekan save. Untuk menutup program klik tombol tanda silang di pojok kanan atas.

4.4 Implementasi Uji Coba

Pada subbab ini akan dilakukan pembahasan mengenai pengujian yang telah dilakukan pada sistem dan hasil evaluasi dari ringkasan hasil sistem. Dalam pengujian ini masukkan seluruh data yang dipelukan dan set parameter algoritma. Penjelasan lebih lengkap mengenai data dan parameter algoritma yang akan di uji.

4.4.1 Hasil Pengujian Fungsionalitas Perangkat Lunak

Pengujian penjadwalan mata kuliah menggunakan algoritma koloni semut dinyatakan berhasil apabila tidak mengalami kegagalan pada proses eksekusi program serta hardconstrain terpenuhi. *Hardconstrain* yang dimaksud adalah dosen tidak boleh mengajar di waktu yang sama, tidak ada bentrok mata kuliah dan pecahan SKS mata kuliah tidak dijadwalkan di jam yang sama. Untuk percobaan pertama jumlah generasinya sebanyak 15, *pheronome* awal sebesar 0,01,α sebesar 1,β sebesar 1 dan penguapan *pheronome* sebesar 0,01. Pada percobaan ke 2 jumlah generasinya sebanyak 10, *pheronome* awal sebesar 0,01, α sebesar 1, β sebesar 5 dan penguapan *pheronome* sebesar 0,01. Pada percobaan ke 3 jumlah generasinya sebanyak 15, *pheronome* awal sebesar 0,01, α sebesar 5, β sebesar 1 dan penguapan *pheronome* sebesar 0,8 Hasil dari pengujian fungsionalitas perangkat lunak dapat dilahat pada tabel 4.1.

Tabel 4.1 Tabel parameter algoritma

Percobaan	Nilai	Dosen	SKS	Semester	Sebelah	area	Waktu
p1	1644	0	0	792	171	14	1539
p2	1609	0	0	714	288	0	2889
p3	1638	0	0	792	209	0	3950

Pada tabel 4.1 dapat dilihat bahwa fungsi penjadwalan berhasil dapat dilihat dari dosen dan SKS yang menunjukkan nilai 0. Itu berarti bahwa *hardconstrain* dari perangkat lunak sudah terpenuhi.

4.4.2 Hasil Pengujian Kinerja Perangkat Lunak

Pengujian kinerja perangkat dilakukan dengan merubah input parameter dari algoritma seperti pada tabel 4.3. Generasi yang digunakan dalam pengujian adalah sebanyak 15 generasi.

Tabel 4.2 Hasil Pengujian 1

No	α	B	Banyak Percobaan			Lintasan		Waktu (s)
			P_1	P_2	P_3	Terbaik	Terburuk	
1	3	0,5	1614	1634	1644	1614	1644	3964
2	3	3	1567	1614	1610	1567	1614	3853
3	0,5	3	1613	1589	1609	1589	1613	4070

Tabel 4.3 Hasil Pengujian 2

No	ρ	Banyak Percobaan			Lintasan		Waktu (s)
		P_1	P_2	P_3	Terbaik	Terburuk	
1	0,3	1615	1615	1615	1615	1615	3781
2	0,5	1555	1575	1590	1555	1590	3785
3	0,8	1620	1630	1623	1620	1630	3815

Tabel 4.4 Hasil Pengujian3

No	τ	Banyak Percobaan			Lintasan		Waktu (s)
		P_1	P_2	P_3	Terbaik	Terburuk	
1	0,3	1592	1611	1612	1592	1612	4244
2	0,5	1651	1626	1589	1589	1651	4393
3	0,8	1580	1589	1585	1580	1589	4304

4.5 Pembahasan

Hasil uji yang didapatkan akan di analisis lebih lanjut untuk melihat apakah hasilnya sesuai dengan tujuan yang ingin dicapai dalam penelitiaan. Selain itu akan diambil kesimpulan terhadap hasil uji yang didapat.

4.5.1 Analisis hasil uji coba fungsionalitas perangkat lunak

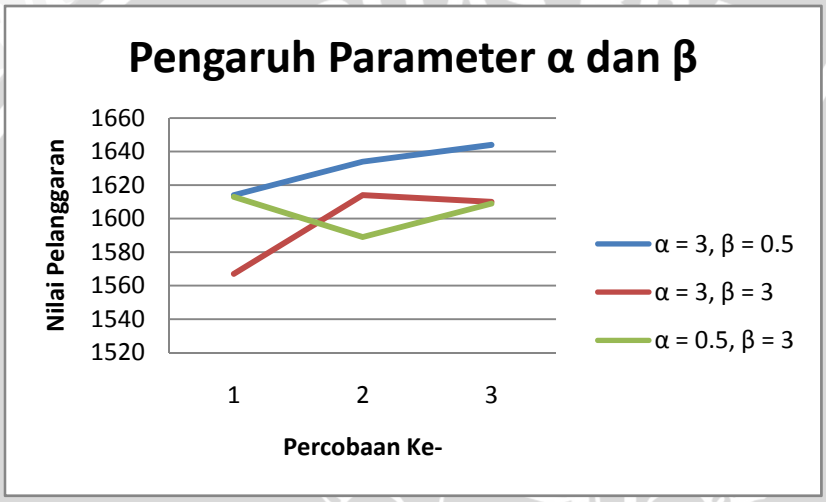
Hasil uji menunjukkan bahwa perangkat lunak penjadwalan matakuliah yang telah dibuat pada penelitian ini telah memenuhi kebutuhan perangkat lunak yang telah dipaparkan pada bab 3. Hal ini dibuktian dengan keberhasilan perangkat lunak

membuat jadwal matakuliah tanpa adanya pelanggaran untuk *hardconstrain*meskipun *softconstrain* masih terdapat pelanggaran.

4.5.2 Analisis hasil uji kinerja perangkat lunak

4.5.2.1 Analisis hasil uji kinerja parameter α dan β

Dalam pengujian parameter α dan β dapat disimpulkan bahwa untuk nilai α dan β sedikit mempengaruhi nilai pelanggaran yang dihasilkan. Nilai α dan β yang terbaik adalah 3 dan 3 menghasilkan nilai terkecil 1567 dan memiliki rata-rata terkecil juga yaitu 1597 itu berati nilai pelanggaran yang dihasilkan lebih kecil dibandingkan dengan nilai pelanggaran parameter yang lainnya. Apabila semakin kecil nilai pelanggaran yang dihasilkan maka sekain kecil pelanggaran jadwal yang dihasilkan oleh jadwal yang dibuat. Dapat dilihat pada gambar 4.15 menunjukkan nilai pelanggaran.



Gambar 4.15 Pengaruh Parameter α dan β

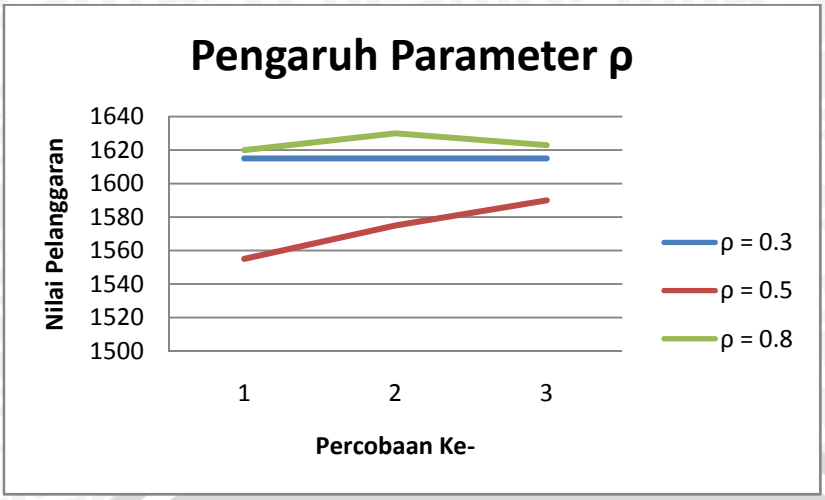
Percobaan	Nilai	Dosen	SKS	Awal	Sebelah	area	waktu
p1	1567	0	0	750	190	0	3853
p2	1614	0	0	816	171	0	4581
p3	1610	0	0	774	209	0	4462

Tabel 4.5 Nilai Pelanggaran parameter α dan β

Pada percobaan yang pertama menunjukkan bahwa nilai pelanggaran total berjumlah 1567 dimana terjadi pelanggaran semester awal dijadwalkan lebih pagi sebanyak 750, pelanggaran pecahan SKS dijadwalkan selang satu hari setelah jadwal pada hari tersebut sebanyak 190 dan pelanggaran untuk dosen yang mengajar berkelanjutan area mengajar berdekatan memiliki nilai pelanggaran sebanyak 0 dan waktu komputasi yang dibutuhkan untuk menyelesaikan proses penjadwalan sebanyak 15 generasi menempuh waktu 3853 detik.

4.5.2.2 Analisa hasil uji kinerja parameter ρ

Pada pengujian parameter α dan βtelah di dapat nilai terbaik untukα dan β adalah 3 dan 3. Untuk melengkapi agar nilai yang dihasilkan semakin baik maka selanjutnya menentukan nilai ρ yang sesuai untuk melengkapi tersebut adalah 0,5. Pada tabel 4.5 terdapat nilai terkecil yaitu 1555 dan memiliki rata-rata terkecil yaitu 1573 dibandingkan dengan nilai parameter yang lainnya. Pada gambar 4.16 menunjukkan nilai pelanggaran.



Gambar 4.16 Pengaruh parameter ρ

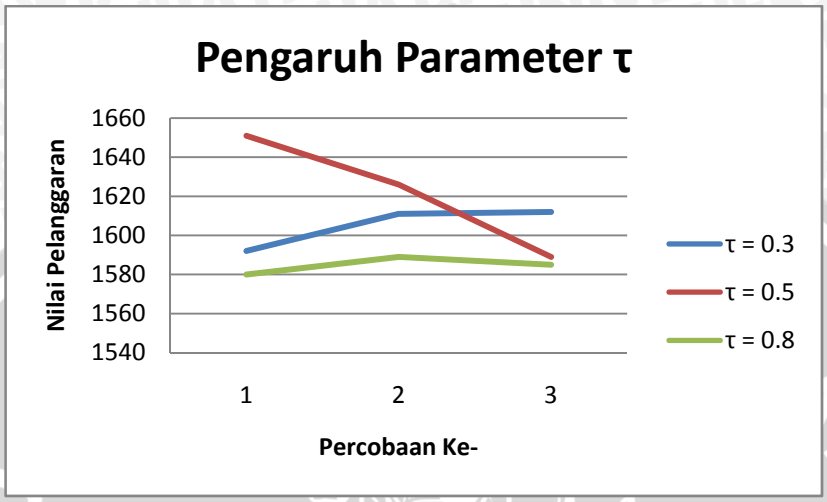
Tabel 4.6 Nilai parameter α dan ρ

Percobaan	Nilai	Dosen	SKS	awal	Sebelah	area	waktu
p1	1555	0	0	720	209	0	3785
p2	1575	0	0	738	209	0	3766
p3	1590	0	0	786	171	7	3786

Pada percobaan yang pertama menunjukkan bahwa nilai pelanggaran total berjumlah 1555 dimana terjadi pelanggaran semester awal dijadwalkan lebih pagi sebanyak 720, pelanggaran pecahan SKS dijadwalkan selang satu hari setelah jadwal pada hari tersebut sebanyak 209 dan pelanggaran untuk dosen yang mengajar berkelanjutan area mengajar berdekatan memiliki nilai pelanggaran sebanyak 0 dan waktu komputasi yang dibutuhkan untuk menyelesaikan proses penjadwalan sebanyak 15 generasi menempuh waktu 3785 detik.

4.5.2.3 Analisa hasil uji kinerja parameter τ

Pada pengujian parameter α dan β telah di dapat nilai terbaik untuk α ,β adalah 3,3dan ρ0,5. Untuk melengkapi agar dapat memberikan nilai yang baik, maka τ yang terbaik adalah 0,01. Pada tabel 4.6 terdapat nilai terkecil yaitu 1555 dan memiliki rata-rata terkecil yaitu 1573 dibandingkan dengan nilai parameter yang lainnya. Pada gambar 4.8menunjukkan nilai pelanggaran.



Gambar 4.17 Pengaruh parameter τ

Tabel 4. 7 Nilai parameter τ

Percobaan	Nilai	Dosen	SKS	Awal	Sebelah	area	waktu
p1	1555	0	0	720	209	0	3785
p2	1575	0	0	738	209	0	3766
p3	1590	0	0	786	171	7	3786

Pada percobaan yang pertama menunjukkan bahwa nilai pelanggaran total berjumlah 1555 dimana terjadi pelanggaran semester awal dijadwalkan lebih pagi sebanyak 720, pelanggaran pecahan SKS dijadwalkan selang satu hari setelah jadwal pada hari tersebut sebanyak 209 dan pelanggaran untuk dosen yang mengajar berkelanjutan area mengajar berdekatan memiliki nilai pelanggaran

sebanyak 0 dan waktu komputasi yang dibutuhkan untuk menyelesaikan proses penjadwalan sebanyak 15 generasi menempuh waktu 3785 detik. Untuk mengetahui jadwal terbaik yang terbentuk dapat dilihat pada lampiran 1.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melakukan penelitian maka disimpulkan :

1. Algoritma koloni semut dapat diterapkan dalam masalah Penjadwalan mata kuliah di Jurusan Matematika Fakultas MIPA Universitas Brawijaya Malang.Berdasarkan penelitian yang telah dilakukan,proses penjadwalan mata kuliahtidak terdapat pelanggaran untuk *hardconstrain*.
2. Pemilihan parameter awal algoritma sangat berpengaruh terhadap waktu komputasi. Parameter awal yang terbaik yaitu $\alpha = 3$ $\beta = 3$ $\rho = 0{,}5$ dan *pheronome* awal0,01. Untuk 5 generasi waktu komputasi yang dibutuhkan sebesar 1539 s, 10 generasi membutuhkan waktu 2889 s, dan untuk 15 generasi membutuhkan waktu sebesar 3950 s.

5.2 Saran

Untuk pengembangan lanjut perangkat lunak maka ada beberapa saran yang dapat diberikan :

1. Pada penelitian ini skala penelitian hanya terbatas pada jurusan. Oleh sebab itu diperlukan untuk memperluas skala penelitian untuk penjadwalan mata kuliah di fakultas.
2. Pada penelitian ini jadwal mata kuliah yang dihasilkan sudah mememenuhi kebutuhan namun waktu yang digunakan untuk menghasilkan jadwal membutuhkan waktu komputasi yang lama dikarenakan titik awal dan titik akhir tidak dapat ditentukan seperti pada problem optimasi rute perjalan.Oleh sebab itu diperlukan suatu metode untuk mempercepat waktu komputsi.

3. Pada penelitian ini untuk mata kuliah wajib dan pilihan tidak dibedakan waktunya. Oleh sebab itu diperlukan pengelompokan mata kuliah wajib dan pilihan dimana mata kuliah wajib dijadwalkan dibawah jam 12.00.

DAFTAR PUSTAKA

Dorigo, Marco.,Caro, G.D dan Luca M. Gambardella. 1999. *Ant Algorithms for Discrete Optimization*. *Artificial Life* Volume 5.Belgian.

Fernandez, A dan E. Handoyo. 2008. *Analyzing Evolutionary Algortihm Method to Optimize Time Table System (Case Study: University Scheduling Time Table)*, Universitas Diponegoro. Semarang.

Kusumadewi, S. dan Purnomo,H. 2005. *Penyelesaian Masalah Optimasi dengan Teknik-teknik Heuristik*. Graha Ilmu.Yogyakarta.hal. 231-386.

Leksono, A. 2009.*Algoritma ant colony optimization (ACO) untuk menyelesaikan masalah traveling salesman problem (TSP). Jurusan Matematika Universitas* Diponegoro.Semarang.

Luknanto, D. 2000. *Pengantar Optimasi Nonlinier*. Jurusan Teknik Sipil Universitas Gajah Mada.Yogyakarta. Hal 2-14

.

Mindaputra, E. 2009.*Penggunaan algoritma ant colony system dalam traveling salesman problem (TSP) pada PT. EKA JAYA MOTOR*. Jurusan Matematika Universitas Diponegoro.Semarang.

Muttakhiroh, I'ing. 2007. *Menentukan Jalur Terpendek Menggunakan Algoritma Semut*. Jurusan teknik informatika Universitas Islam Indonesia.Yogyakarta.

Pinedo, M. L., 1994. *Schedulling Theory, Algorithms and Systems*. Springer. New York. hal. 1-6

Wardy, I.S. 2007.*Penggunaan Graf dalam Algoritma Semut untuk Melakukan Optimisasi.(online)*, (http://elesys.fsaintek.unair.ac.id, diakses 15 Oktober 2011)

Weise,T. 2009. *Global Optimization Algorithms-Theory and Application*.(online),(http://www.itweise.de/projects/book.pdf,Tanggal akses: 13 Oktober 2011).

Yang, F. C dan Yu H.H. 2008.*Ant Colony Optimization Method for Time Window Constrainted Batching and Scheduling Problem*.APIEMS.Bali.

Zhang, Jun , X. Hu dan J.H Zhong. 2006. *Implementation of ant Colony Optimazition technique for job shop scheduling problem*. Sun Yat-sen University. PR China.

Zukhri,Zdan Shidiq, A.2004.*Algoritma Semut pada penjadwalan produksi Jobshop*. Universitas Islam Indonesia. Yogyakarta.

LAMPIRAN