KOMPARASI METODE *ANT COLONY OPTIMIZATION* DENGAN TABU *SEARCH* UNTUK PENJADWALAN PERKULIAHAN

Komang Ayu Triana Indah
Jurusan Teknik Elektro Politeknik Negeri Bali, Bukit Jimbaran, Badung ,Bali – 80364 Telp. (0361) 701981
e-mail:triana_indah@pnb.ac.id

Abstrak: Masalah penjadwalan kuliah merupakan masalah yang sangat kompleks. Inti penjadwalan tersebut adalah menjadwalkan beberapa komponen yang terdiri dari mahasiswa, dosen, ruang, waktu, dan matakuliah dengan memerhatikan sejumlah batasan dan syarat (*constraint*) tertentu. Setiap langkah yang dilakukan oleh *Tabu Search* diambil berdasarkan hasil perhitungan *cost* yang dilakukan tiap iterasi untuk memilih *neighbour solution* yang akan menjadi *current best solution* berikutnya. Sedangkan ACO (Ant Colony Optimization) menggunakan algoritma yang diadaptasi dari prilaku semut untuk menyelesaikan permasalahan kombinatorial. Penelitian ini dirancang untuk membandingkan unjuk kerja metode *Tabu search* dan ACO (*Ant Colony Optimization*) melalui aplikasi penjadwalan perkuliahan dengan menggunakan metode ACO dan Tabu Search didesain dengan memasukkan beberapa parameter yaitu data dosen, matakuliah, ruangan, dan beberapa variabel dari masing-masing parameter yang kemudian diproses sehingga menghasilkan penjadwalan perkuliahan. Adapun hasil yang diperoleh dari penggunaan Tabu Search dan ACO (Ant Colony Optimization) untuk memecahkan masalah penjadwalan kuliah dan ujian di perguruan tinggi. Hasil penelitian untuk membandingkan unjuk kerja antara kedua metode dilihat dari jumlah *constrain* yang terlanggar serta lamanya waktu yang diperlukan dari masing-masing iterasi pada tiap metode sampai mendapatkan jadwal kuliah.

Kata Kunci : Ant Colony Optimization, Tabu Search,constraint, penjadwalan, cost, iterasi.

Comparison of Ant Colony Optimization Method With Tabu Search For Scheduling Lectures

Abstract: *Scheduling Problem Subject is a very complex issue, where the core of the scheduling is how to schedule multiple components consisting of students, faculty, space, time, and subject to consider a number of restrictions and requirements (constraints) specific. Every step taken by Tabu Search was based on calculations performed cost per iteration to choose neighbor solution that will be current next best solution. While the ACO algorithm is adapted from the behavior of ants to solve combinatorial problems. This study was designed to compare the performance of the method Tabu search and ACO through the application of scheduling lectures using the ACO and Tabu Search is designed to include some parameters of data Lecturer, Subject, rooms, and some of the variables of each parameter are then processed to produce scheduling lectures. The results obtained from the use of Tabu Search and ACO to solve the problem of scheduling lectures and exams in PerguruanTinggi. The results of the study to compare the performance between the two methods seen from the constraint is violated and the length of time it takes from each iteration of each method to obtain class schedules.*
Keywords*: Ant Colony Optimization, Tabu Search , constraints, scheduling, cost, iteration.*

1. PENDAHULUAN

Heuristik berasal dari kata Yunani *heuriskein* yang berarti seni untuk menemukan strategi dalam menyelesaikan persoalan, sedangkan *meta* berarti metodologi tingkat tinggi atau lanjut (Talbi, 2009). Di dalam ilmu komputer, metode heuristik merupakan suatu teknik untuk penyelesaian permasalahan yang tidak menekankan pada pembuktian apakah solusi yang didapatkan adalah benar karena pembuktian, apakah suatu solusi benar merupakan fokus dari metode penyelesaian analitik. Metode heuristik merupakan suatu metode penyelesaian yang menggunakan konsep pendekatan.

Masalah penjadwalan secara umum adalah aktivitas penugasan yang berhubungan dengan sejumlah kendala, sejumlah kejadian yang dapat terjadi pada suatu periode waktu dan tempat/lokasi tertentu, sehingga fungsi objektif sedekat mungkin terpenuhi. Masalah ini muncul di berbagai bidang kegiatan maupun instansi seperti: rumah sakit, universitas, penerbangan, pabrik, dan lain-lain. Desain model masalah penjadwalan bervariasi sesuai dengan kebutuhan serta keadaan di lapangan.

Penyampaian informasi pada lembaga akademik merupakan hal yang sangat penting terutama informasi yang berkaitan dengan kegiatan perkuliahan. Salah satunya adalah informasi jadwal kuliah. Penjadwalan kuliah merupakan pekerjaan yang tidak mudah. Inti masalah ini adalah menjadwalkan berbagai komponen yang terdiri dari mahasiswa, dosen, ruang,

dan waktu dengan memperhatikan sejumlah batasan dan syarat tertentu.

1.2 Masalah

Terdapat beberapa permasalahan dalam penjadwalan mata kuliah di antaranya:

1. Pada bagian Jurusan/Program Studi, yaitu: bagaimana mengidentifikasi matakuliah yang ditawarkan sesuai dengan kurikulum pada semester yang akan dibuat penjadwalan?
2. Bagaimana menyusun jadwal perkuliahan bagi mahasiswa maupun dosen meliputi penentuan matakuliah, waktu, dan tempat perkuliahan serta penentuan dosen pengampu setiap mata kuliah?
3. Bagaimana melakukan penjadwalan kuliah yang dilakukan oleh jurusan yang berisi nama hari dan jam kuliah, matakuliah, ruangan dan nama dosen secara sistematis sehingga tidak terjadi bentrok?
4. Menentukan syarat minimal 16 kali tatap muka antara dosen dan mahasiswa selama satu matakuliah dalam setiap semester, atau minimal 75% perkuliahan sebelum mengikuti ujian.
5. Bagaimana menentukan penjadwalan perkuliahan sehingga beban sks setiap dosen minimal 12 sks persemester?

Syarat-syarat dalam penjadwalan kuliah terbagi dalam dua kelompok sesuai dengan tingkat kewajiban syarat tersebut terpenuhi, yaitu: *hard constraint* (harus terpenuhi) dan *soft constraint* (diupayakan untuk terpenuhi). Sebuah solusi hanya dapat dikatakan sah dan *valid* apabila dalam solusi tersebut sama sekali tidak ada *hard constraint* yang terlanggar. Berbeda dengan *hard constraint*, kendala yang termasuk dalam *soft constraint* adalah kendala yang tidak selalu dapat terpenuhi dalam proses pembentukan jadwal. Akan tetapi, meskipun tidak harus terpenuhi, jadwal yang dihasilkan harus semaksimal mungkin berusaha memenuhi ketentuan *soft constraint*. Masalah penjadwalan kuliah dapat diselesaikan dengan beberapa metode heuristik, seperti: *tabu search, simulated annealing*, dan algoritma genetika [1]. Dalam penelitian ini, permasalahan penjadwalan kuliah akan diselesaikan dengan menggunakan algoritma *algoritma Ant Colony Optimation (ACO)* yang selanjutnya dibandingkan dengan metode *Tabu Search*, sehingga dengan digunakannya algoritma ini diharapkan akan diperoleh hasil penjadwalan yang lebih optimal. Pada penelitian ini dirancang sebuah aplikasi untuk mengatur jadwal kuliah secara otomatis agar menghasilkan luaran berupa jadwal kuliah, jadwal ini masih memungkinkan untuk diubah secara manual sesuai dengan keinginan dari *user*. Untuk pembuatan aplikasi akan digunakan sebuah algoritma *metaheuristic* yaitu *Ant Colony* atau yang lebih dikenal dengan nama algoritma semut yang kemudian dibandingkan kinerjanya dengan algoritma *Tabu Search* untuk mendapatkan hasil yang lebih optimal.

1.3 Tujuan

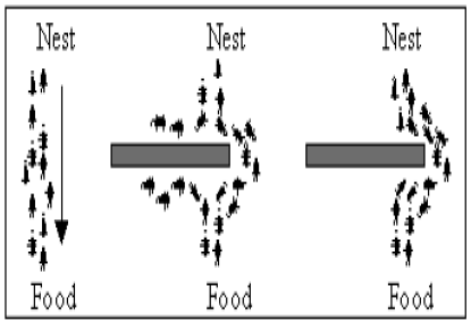
Adapun tujuan penelitian ini yaitu:

1. Merancang aplikasi penjadwalan perkuliahan dengan metode ACO yang dibandingkan dengan metode *Tabu Search.*
2. Untuk mengevaluasi aplikasi *scheduling* dengan metode *ACO* dan *Tabu Search.*
3. Menganalisis optimalisasi algoritma semut ini dimulai dari penggunaan tipe data, waktu kalkulasi, perbandingan nilai *constraints* dan efektivitas perulangan yang dilakukan.
4. Melakukan pengujian dan menganalisis hasil pengujian sistem serta mengenai penggunaan algoritma ACO, mulai dari alasan penggunaan algoritma, cara penyelesaian hingga optimasi yang dilakukan oleh algoritma ACO yang dibandingkan dengan metode *Tabu Search* pada aplikasi penjadwalan perkuliahan.

2. TINJAUAN PUSTAKA

2.1 Metaheuristik

Menurut Talbi (2009), metaheuristik dapat didefinisikan sebagai metode lanjut (*advanced*) berbasis heuristik untuk menyelesaikan persoalan



Gambar 1. Ants menemukan jalur terpendek disekeliling *Obstacle* (Dorigo, 1996*)*

optimisasi secara efisien. Di dalam *wikipedia*, metaheuristik didefinisikan sebagai metode optimisasi yang dilakukan dengan memperbaiki kandidat penyelesaian secara *iteratif* sesuai dengan fungsi objektifnya. Metode ini mampu menghasilkan penyelesaian yang baik dalam waktu yang cepat (*acceptable*), tetapi tidak menjamin bahwa penyelesaian yang dihasilkan merupakan penyelesaian terbaik (optimal). Metode metaheuristik banyak dipakai dalam optimisasi stokastik (optimisasi

stokastim merupakan optimisasi yang memiliki derajat ketidakpastian atau random).

1. Terbatas di ruang pencarian.
2. Konsep dasar dari metaheuristik memungkinkan pendeskripsian secara abstrak
3. Metaheuristik bersifat *general* sehingga dapat diterapkan dalam berbagai macam persoalan
4. Metaheuristik dapat menggunakan domain pengetahuan khusus dalam bentuk heuristik yang dikendalikan dengan strategi tingkat lanjut
5. Metaheuristik dapat menggunakan pengalaman yang didapat selama proses pencarian untuk menuntun proses pencarian.

2.2 *Ant Colony Optimization (ACO)*

ACO (*Ant Colony Optimization*) atau Algoritma Koloni Semut adalah sebuah probabilistik komputasi teknik untuk memecahkan masalah yang dapat dikurangi untuk menemukan jalur yang baik melalui grafik. Algoritma pertama yang bertujuan untuk mencari jalan yang optimal dalam grafik, berdasarkan perilaku semut mencari jalan antara koloni dan sumber makanan. Asli ide ini telah diversifikasi untuk menyelesaikan kelas yang lebih luas dari masalah numerik, dan sebagai hasilnya, beberapa masalah telah muncul, menggambar tentang berbagai aspek perilaku semut. ACO pertama kali diperkenalkan oleh Marco Dorigo, ACO itu sendiri terinspirasi oleh koloni-koloni semut dalam mencari makan. Semut-semut tersebut meninggalkan zat (*pheromone*) di jalan yang mereka lalui.

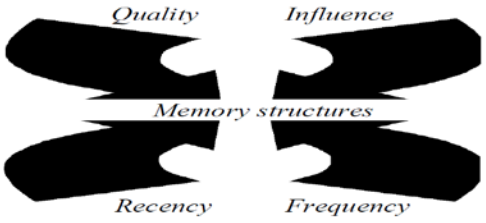
ACO adalah sistem agen yang menyimulasikan tingkah laku natural dari kelompok semut (*ants*) yang mengandung mekanisma kerjasama dan adaptasi. Ide dasar dari proses ini dapat dilihat pada Gambar 1.

Algoritma ini terinspirasi dari tingkah laku koloni semut dalam mencari makan, Di dunia nyata, semut (awalnya) berjalan secara acak, dan ketika menemukan makanan kembali ke koloni mereka sambil meletakkan *pheromone* jejak. Jika semut lain menemukan jalur tersebut, mereka tidak cenderung untuk menjaga bepergian secara acak, tetapi malah mengikuti jejak, kembali dan menguatkannya jika pada akhirnya mereka menemukan makanan..

Gagasan awalnya berasal dari mengamati makanan eksploitasi sumber daya di antara semut, di mana semut 'secara individual memiliki kemampuan kognitif terbatas secara kolektif mampu menemukan jalur terpendek antara sumber makanan dan sarang. Semut pertama menemukan sumber makanan (F), melalui cara apapun (a), kemudian kembali ke sarang (N), meninggalkan jejak *pheromone* (b) Semut tanpa pandang bulu cara mengikuti empat kemungkinan, tapi penguatan landasan membuatnya lebih menarik sebagai rute terpendek.

2.3 Tabu Search

Tabu search pertama kali diperkenalkan oleh Glover sekitar tahun 1986. Glover menyatakan bahwa *tabu search* adalah salah satu prosedur metaheuristik tingkat tinggi untuk penyelesaian permasalahan optimasi kombinatorial (Glover, 1986). *Tabu search* dirancang untuk mengarahkan metode-metode lain (atau komponen proses *tabu search* itu sendiri) untuk keluar atau menghindari dari masuk dalam solusi optimal yang bersifat lokal. Kemampuan *tabu search* dalam menghasilkan solusi yang mendekati optimal telah dimanfaatkan dalam beragam permasalahan berbagai bidang seperti penjadwalan hingga bidang telekomunikasi. Pemilihan kandidat terbaik didasarkan pada nilai fungsi tujuan. Pemeriksaan nilai fungsi tujuan terlebih dahulu dilakukan sebelum pemeriksaan status *tabu*. Apabila nilai fungsi tujuan sebuah kandidat lebih baik dari yang lain, maka kandidat tersebut berpotensi untuk diterima sehingga perlu diperiksa status *tabu*-nya. Pemilihan kandidat solusi terbaik yang dilakukan oleh *tabu search* menggunakan prinsip *global test strategy* bukan *first best strategy*. Sedangkan *first best strategy* adalah strategi dimana algoritma akan mengganti solusi terbaik saat ini secara langsung jika solusi yang lebih baik ditemukan. Pemilihan kandidat solusi dalam *tabu search* juga tidak dilakukan secara probabilistik sebagaimana *ant colony system*, *simulated annealing* dan *genetic algorithm*. Karakteristik ini menjadikan solusi yang dihasilkan *tabu search* akan sama setiap kali dilakukan proses pencarian solusi terhadap suatu permasalahan.
dan *influence* yang ditunjukkan oleh gambar 2.4 (Gendreau et.al, 1998):



Gambar 2.Dimensi Tabu Search

Pada tiap iterasi, dipilih solusi baru yang merupakan solusi terbaik dalam *neighbourhood* dan tidak tergolong sebagai *tabu*. *Global-best strategy* adalah strategi dimana algoritma akan mengganti solusi terbaik saat ini dengan solusi terbaik yang ada pada *neighborhood*. Algoritma *tabu search* memiliki kemampuan untuk keluar dari solusi optimum lokal,

tetapi *tabu search* tidak dapat menentukan optimum global. Semakin besar jumlah maksimum iterasi, semakin besar pula peluang untuk menentukan solusi optimal secara global, namun memerlukan waktu perhitungan yang lama.

Secara garis besar elemen-elemen utama pada *tabu search* adalah sebagai berikut:

1. Representasi solusi: setiap solusi *feasible* pada suatu permasalahan optimasi harus direpresentasikan secara unik.
2. Fungsi *cost*: setiap fungsi *cost* (fungsi tujuan) akan memetakan setiap solusi *feasible* ke nilai *cost*-nya.
3. *Neighbourhood* (tetangga): suatu fungsi *cost* akan memetakan setiap solusi *feasible S* ke solusi-solusi yang lainnya.
4. *Tabu list*: suatu *list* yang berisi T gerakan (*move*) terakhir.
5. Jumlah elemen yang harus ada pada suatu solusi.

Hal-hal di atas dapat dikatakan sebagai bahan-bahan dasar dari metode *tabu search* yang nantinya akan digunakan untuk memecahkan masalah penjadwalan kuliah pada pembahasan selanjutnya.

Algoritma Tabu Search

Algoritma *tabu search* secara garis besar dapat ditulis sebagai berikut:

Langkah 0. Tetapkan:
X = Matriks input berukuran nxm.
MaxItr = maksimum iterasi.

Langkah 1 Inisialisasi awal
S = bangkitkan solusi secara random ini dilakukan untuk mendapatkan *Candidate List* yang pertama. Dari *candidate list* yang pertama kita dapatkan GlobalMin = F Cost (S) yang pertama.
Kemudian Best = S.

Langkah 2 Selanjutnya S dimasukan ke dalam TabuList = [ ].

Langkah 3 Kerjakan dari k = l sampai MaxItr:
BestSoFar = FCost (S).
BestMove = S.

Langkah 4 Kerjakan dari i = l sampai (n-l):
Kerjakan dari j = I sampai n:
L = Tukar (S [I], 5 [j]).
Cost = FCost (L)

Langkah 5 Kemudian dilakukan Tabu test. Jika (L □ T~buList) atau (Cost < GlobalMin), kerjakan.

Langkah 6 Kemudian dilakukan Aspiration test.
Jika (Cost < BestSoFar),
kerjakan :
BestSoFar = Cost.
BestMove = L.
S = BestMove.

Langkah 7 Tambahkan S ke TabuList. Disini Tabu List di Update
Jika BestSoFar < GlobalMin, kerjakan:
GlobalMin = BestSoFar.
Best = BestMove.
Disini Best Solution di Update.

Langkah 8 Completion check, disini di check apakah semua iterasi sudah dilakukan.

Langkah 9 Selesai. Solusi akhir adalah Best, dengan cost sebesar GlobalMin.

3.METODE PENELITIAN

3.1 Rancangan Umum Penelitian

Pada perancangan awal ditentukan terlebih dahulu diagram suatu penjadwalan untuk memudahkan dalam penyusunan flowchart algoritma programnya, diantaranya:

a. Diagram *Use Case*

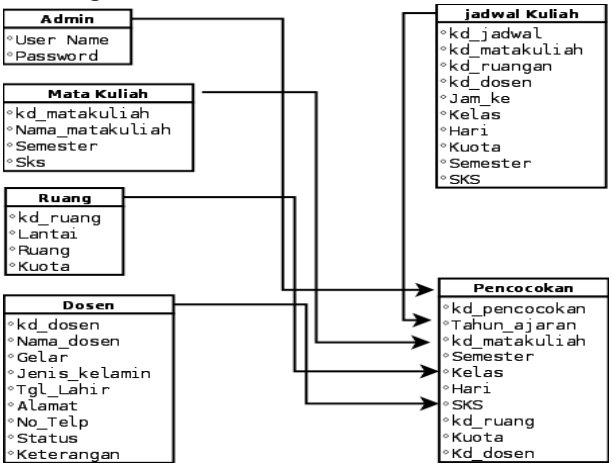
Diagram *use case* adalah sebuah diagram yang digunakan untuk menunjukkan tampilah grafis dari fungsionalitas yang diberikan oleh sistem dilihat dari sisi aktor, tujuan aktor, dan hal yang berkaitan dengan *use case* yang ada.

b. Diagram Aktivitas

Diagram aktivitas adalah sebuah teknik penjelasan diagram yang bebas menunjukkan aliran aktivitas dan kegiatan langkah demi langkah. Diagram pada gambar 3, biasanya digunakan untuk menjelaskan aliran bisnis dan operasi dari sebuah komponen dari sistem .

c. Diagram Kelas

Diagram ini diwakilkan oleh diagram kelas yang merupakan sebuah tampilan statis struktur program. Diagram kelas menggambarkan struktur dengan memperlihatkan kelas dari sistem, baik atribut maupun relasi dengan kelas lain.



Gambar 3. Relasi antar Tabel

3.2 Algoritma ACO

Algoritma ACO didasarkan pada ide-ide berikut:

Masing-masing jalur (*path*) yang diikuti oleh semut diasosiasikan sebagai kandidat solusi. Ketika seekor semut melalui sebuah jalur, sejumlah dijatuhkan pada jalur sesuai dengan kualitas term (hubungan) kandidat solusi untuk *"Target problem"* Kaidah yang terbaik yang dibangun oleh seluruh semut dianggap sebagai kaidah yang dicari. Kaidah yang lain dibuang. Hal ini melengkapi sebuah iterasi dari sistem itu. Algoritma ini menggunakan kontruksi kaidah dan Peningkatan *pheromone*.

ACO itu sendiri terinspirasi oleh koloni-koloni semut dalam mencari makan. Semut-semut tersebut meninggalkan zat (*pheromone*) di jalan yang mereka lalui. Algoritma ACO ini merupakan algoritma pencarian berdasarkan probabilistik, di mana probabilistik yang digunakan merupakan probabilistik dengan bobot sehingga butir pencarian dengan bobot yang lebih besar akan berakibat memiliki kemungkinan terpilih yang lebih besar pula.

Dari gambar di atas maka *Pheromone* ACB> *pehromone* ADB

Algoritma umum ACO.

masukan data permasalahan
while not selesai do
bangkitkan semut
bangkitkan solusi
update pheromone
hancurkan semut
end while
keluarkan solusi

3.3 Tabu Search untuk Sistem Penjadwalan Kuliah

Implementasi metode Tabu Search untuk penjadwalan mata kuliah mengikuti beberapa tahapan sebagai berikut.

a. Klasifikasi Constraint

Peraturan-peraturan yang dimiliki oleh suatu Perguruan Tinggi dalam pembuatan jadwal kuliah dapat dirangkum dan dikelompokan menjadi 2 kelompok sesuai dengan tingkat kewajiban peraturan itu dipenuhi. Kedua kelompok ini diuraikan pada bagian *Hard Constraint* dan *Soft Constraint*.

1. Hard Constraint

Hard Constraint didefinisikan sebagai *constraint* wajib yang harus dipenuhi dalam proses perhitungan algoritma. Sebuah solusi hanya dapat dikatakan sah dan valid apabila dalam solusi tersebut sama sekali tidak ada *hard constraint* yang terlanggar. Berikut adalah daftar *Hard constraint* dalam masalah penjadwalan kuliah beserta penjelasannya :

1. Jadwal dosen tidak boleh bentrok.
2. Ruangan yang digunakan tidak boleh bentrok.
3. Dosen harus mengajar matakuliah yang sesuai dengan ketentuan dari Ketua Jurusan.
4. Dosen tidak boleh mendapatkan jadwal mengajar melebihi jatah SKS-nya.
5. Seluruh kelas yang dibuka harus mendapatkan waktu dan ruangan perkuliahan.
6. Jumlah mahasiswa dalam satu kelas tidak boleh melebihi kapasitas ruangan.
7. Kelas matakuliah yang membutuhkan ruangan khusus harus dijadwalkan di ruangan yang tepat.
8. Kelas-kelas matakuliah yang lebih dari 1 pertemuan per minggu harus mendapat jadwal yang berurutan.

2. Soft Constraint

Berbeda dari *hard constraint*, kendala yang termasuk dalam kategori *soft constraint* adalah kendala yang tidak selalu dapat terpenuhi dalam proses pembentukan jadwal kuliah. Meskipun harus tidak terpenuhi, tetapi jadwal kuliah yang dihasilkan harus semaksimal mungkin berusaha memenuhi ketentuan *soft constraint* ini. Berikut adalah daftar *soft constraint* dalam masalah penjadwalan kuliah, beserta penjelasannya :

1. Untuk setiap kelas yang dibuka diusahakan ada dosen yang mengajar.
2. Seorang dosen tidak boleh mendapatkan jadwal mengajar lebih dari 3 pertemuan dalam sehari.
3. Seorang dosen diusahakan tidak mendapat jadwal mengajar hanya 1 pertemuan dalam sehari (bagi dosen yang jumlah sks mengajar lebih dari 2).
4. Jadwal mengajar dosen dalam sehari diusahakan urut tanpa jeda.
5. Jadwal kuliah mahasiswa dalam sehari diusahakan urut tanpa jeda.
6. Jadwal mengajar dosen dalam sehari diusahakan di dalam 1 lokasi kampus.
7. Jadwal kuliah mahasiswa paket dalam sehari diusahakan di dalam satu lokasi kampus.
8. Jumlah mahasiswa yang dialokasikan ke dalam satu ruangan kelas diusahakan sedekat mungkin dengan kapasitas ruangan tersebut
9. Kelas-kelas matakuliah yang sama diusahakan untuk terjadwal di hari dan jam yang berbeda.
10. Dosen harus mendapatkan jadwal mengajar sesuai dengan waktu kesediaan mengajarnya.

b. Inti Algoritma

Prinsip dasar dari algoritma *Tabu Search* adalah memproses sebuah *empty initial solution* hingga menjadi *final solution*. Algoritma ini menggunakan hubungan kelas dan matakuliah sebagai variabel serta menggunakan dosen, matakuliah, waktu, dan ruangan sebagai *value* yang akan dimasukkan kedalam variabel yang ada. Setiap langkah yang dilakukan oleh *Tabu Search* diambil berdasarkan hasil perhitungan *cost* yang dilakukan tiap iterasi untuk memilih *neighbour solution* yang akan menjadi *current best solution* berikutnya. Penghentian proses terjadi apabila jumlah

kelas yang terjadwal telah terpenuhi syarat atau interaksi dari user menghendaki penghentian proses.

c. Perhitungan Cost

Perhitungan *Cost* dalam *Tabu Search* adalah suatu proses yang sangat penting. Melalui perhitungan *cost* ini *Tabu Search* dapat mendeteksi pelanggaran *constraint* pada masing-masing kelas. Pendeteksian pelanggaran ini dilakukan untuk menentukan kelas mana yang harus diperbaiki, kemudian memilih dosen dan jam kuliah yang harus dipilih untuk memperbaiki jadwal kelas tersebut. Dapat dikatakan keberhasilan *Tabu Search* dalam menentukan solusi yang mempunyai tingkat kebenaran yang tinggi tergantung dari cara yang digunakan untuk menentukan *cost* dari suatu kelas. Perhitungan cost pada algoritma *Tabu Search* telah dirancang untuk menghasilkan nilai 0, apabila kelas tersebut sudah memiliki komposisi dosen dan waktu kuliah yang paling baik, yang tidak melanggar semua *constraint* yang ada. Semakin tinggi *cost*, dapat disimpulkan bahwa jadwal yang dimiliki kelas tersebut mengandung banyak pelanggaran *soft constraint.* Agar solusi yang diinginkan tercapai, tiap pelanggaran *constraint* didefinisikan sebagai nilai *cost* yang berbeda-beda. Hal ini dilakukan karena tingkat keharusan untuk memenuhi sebuah *soft constraint* berbeda satu sama lain. *Soft Constraint* yang lebih perlu untuk dipenuhi diberi bobot yang lebih tinggi daripada *soft constraint* yang dipandang lebih bisa toleransi.

3.4 Data Masukan

Untuk proses masukan data, terdiri dari :

a. Proses pemasukan data yang berupa data dosen, data matakuliah, data ruang, dan data waktu kuliah. Data dosen diisi oleh Dosen sendiri dengan mengisi formulir data mengajar, sedangkan ruangannya ditentukan oleh Sekretariat Program, bagian penjadwalan kuliah dan ujian untuk disimpan dalam *database.*

b. Proses pemasukan data matakuliah yang ditawarkan tiap semesternya, termasuk penentuan jumlah kelas per-matakuliah yang ditawarkan yang disesuaikan dengan kesanggupan dosen mengajar.

3.5 Data Keluaran

Data keluaran yang dihasilkan berupa:

a. Laporan (*print out*) jadwal kuliah, yang berisi data mata kuliah perjurusan yang diadakan tiap semester. Laporan ini selanjutnya digunakan mahasiswa untuk *key-in* kuliah.

b. Laporan (*print out*) jadwal dosen, merupakan laporan mengajar dosen yang diserahkan kepada dosen yang bersangkutan agar mengetahui jadwal mengajarnya. Sedangkan jadwal ujian diberikan satu minggu sebelum ujian dimulai

3.6 Perancangan Perangkat Lunak

Metode perancangan yang digunakan dalam pembuatan program simulasi Penjadwalan Kuliah menggunakan ACO dan *Tabu Seach* yaitu sebuah algoritma yang dalam teknik penyelesaiannya menggunakan kedua metode untuk dibandingkan sehingga mendapatkan hasil penjadwalan yang optimal. Metode ini menggunakan alat-alat pengembangan yang berupa data *flow diagram* (DFD), diagram relasi antartabel, dan *flowchart* (diagram alir).

3.7 Rancangan Antarmuka

Antarmuka dari aplikasi penjadwalan kuliah dan ujian dengan menggunakan metode Tabu Search terbagi menjadi tiga bagian, yaitu :

3.7.1 Menu Input Data

1.Input Data Dosen

Menu ini digunakan untuk pemasukan data dosen yang mengajar, baik dosen tetap atau dosen tidak tetap.

2. Input Data Ruangan

Menu ini digunakan untu k pemasukan data ruangan yang digunakan.

3. Input Data Matakuliah

Menu ini digunakan untuk pemasukan data Matakuliah, baik semester ganjil atau semester genap.

4.Input Data Match/Pencocokan

Menu ini digunakan untuk mencocokan data input dengan data permintaan mengajar dosen.

5.Menu Proses Penjadwalan dengan *Tabu Search* dan ACO,

Menu ini digunakan untuk memproses penjadwalan kuliah.

4 Menu About, menu ini digunakan untuk pemanggilan antarmuka tentang pembuat sistem dan pertolongan seputar penggunaan sistem.

5 Menu Exit

Menu ini digunakan untuk keluar dari sistem.

4.HASIL DAN PEMBAHASAN

Perangkat lunak (*software*) komparasi metode *ant colony optimization* dengan *tabu search* pada Penjadwalan Kuliah diimplementasikan dengan Microsoft Visual Basic 6.0. Implementasi sistem merupakan tahap sistem siap untuk dioperasikan pada keadaan yang sebenarnya, sehingga akan diketahui sistem yang dibuat benar-benar dapat menghasilkan tujuan yang diinginkan. Sebelum program diterapkan dan diimplementasikan, maka program harus bebas dari kesalahan (*error free*). Kesalahan program yang mungkin terjadi antara lain kesalahan penulisan bahasa, kesalahan sewaktu proses atau kesalahan logikal. Setelah program bebas dari kesalahan, program diuji coba dengan memasukkan data untuk diolah.

Sistem perangkat lunak yang akan dibuat adalah sebuah perangkat lunak yang berisi banyak data input, perhitungan menggunakan *Tabu Search* dan

ACO, peraturan yang ada pada *Tabu Search*, pola desain dan pemakaian pada lingkup kerjanya, dan fleksibilitas. Maka dibutuhkan sebuah bahasa pemrograman yang terstruktur, handal, praktis, mudah digunakan dan juga mendukung batasan sistem operasi secara umum. *Visual Basic* adalah salah satu bahasa pemrograman yang memiliki semua persyaratan yang dibutuhkan dalam proses pembuatan perangkat lunak ini.

4.1 Penjadwalan dengan Algoritma Tabu Search

Dibuat sebuah tabel Pencocokan (*Tabu List*,sebagian). Dari tabel tersebut ada beberapa kode pencocokan yang dijadikan *Local Solution* untuk *Tabu Search* dan diletakkan pada *Tabu List* (Database kode Pencocokan). Nilai 0 merupakan *Global Min*, dan merupakan nilai Cost terbaik. Pada tabel didapatkan penjadwalan dengan metode Tabu Search untuk periode tahun 2015/2016, semester ganjil dan genap. Berdasarkan perhitungan, iterasi didapatkan dari jumlah hari dalam satu minggu yaitu 6,

Total pelanggaran *constraint* =227

Total Global min = 2638

Dari hasil penjadwalan *Tabu Search* dapat disimpulkan:

a) *Final solution* dengan nilai *constraint* terkecil yaitu pada saat Nilai Constraint 0 (tidak ada pelanggaran), dan saat Nilai Global min =1.

b) Hasil terbaik dilihat dari Nilai *Global Min/Best So Far* yaitu pada saat
Iterasi 1, Kelas 203, *Constraint* =0, *Global Min* = 1
Iterasi 1, Kelas 202, *Constraint* =0, *Global Min* = 1
Iterasi 1, Kelas 102, *Constraint* =0, *Global Min* = 1
Iterasi 1, Kelas 102, *Constraint* =0, *Global Min* = 1

c) Nilai Constraint terbesar yaitu: **4**, dengan *Global min* = **45,** yaitu saat
Iterasi 5, Kelas 204, *Constraint* = 4, *Global Min* = 45

Dari tabel juga terlihat jika efektifitas penjadwalan tabu search lebih dominan pada penjadwalan Semester Ganjil. Namun urutan hari pada tabel menjadi acak disebabkan pada saat proses Iterasi berlangsung, diutamakan pencarian nilai Cost dan *Global min* terbaik, jadi faktor urutan hari dan *time slot* tidak terlihat secara berurutan.

4.2 Penjadwalan dengan menggunakan Algoritma ACO

Perhitungan dengan menggunakan algoritma ACO yaitu sebagai berikut:

Diasumsikan parameter yang digunakan pada penjadwalan ini adalah:

1. Hari yang digunakan adalah enam hari dan masing-masing hari terdiri atas 15 *timeslot*, dimana 1 timeslot = 45 menit, sehingga total timeslot selama 1 minggu yaitu: 15 x 6 =90 timeslot.
2. Mata kuliah yang dijadwalkan sebanyak 28 dan ruangan sebanyak 16 kelas
4. Node diasumsikan ruangan kelas yang akan digunakan untuk menjadwalkan mata kuliah berdasarkan *timeslot* yang sudah ditentukan.

Pada penelusuran graf bertujuan untuk menentukan matakuliah yang sesuai dengan kelas agar tidak bentrok. Jumlah ruangan yang dipergunakan yaitu 16 kelas, jumlah hari perkuliahan yaitu 6 hari, dan time slot 90.

5. Hari yang digunakan untuk penjadwalan adalah Senin s/d Jumat dimana Timeslot/hari = 15, satu timeslot = 45 menit, 45 x 15 = 7 jam 15 menit. Diasumsikan penjadwalan disusun dari pukul 07.00 s/d 14.15.

Dari data di atas dapat disimpulkan:

1. Bahwa iterasi terbaik adalah iterasi dari koloni 3, Kelas DH-203, nilai pelanggaran *Constraint* = 0, *Timeslot* = 032, dengan *timeslot* terkecil diasumsikan durasi waktu yang diaplikasikan paling pendek.

2. Hasil pelanggaran *constraint* terkecil tersebut cenderung diperoleh bukan pada iterasi pertama, karena prinsip dari ACO, update *pheromone* terjadi pada iterasi-iterasi besar sehingga constraint bisa terdeteksi tidak pada saat iterasi pertama.

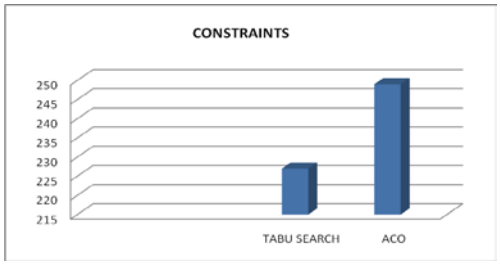
3. Total pelanggaran *constraint* dari penjadwalan ACO yaitu 249

4. Total timeslot yang digunakan dalam keseluruhan penjadwalan yaitu 30062.

4.3 Grafik perbandingan Penjadwalan Tabu Search dan ACO

4.3.1 Constraint yang dilanggar

Grafik perbandingan Penjadwalan Tabu Search dan Ant Colony Optimization dilihat dari constraint yang dilanggar dapat dilihat pada gambar 4.berikut:



Gambar 4.Perbandingan Tabu Search dan ACO dari pelanggaran constraint

Dari grafik dapat dilihat bahwa hasil penjadwalan ACO memiliki nilai constraint lebih besar daripada *Tabu Search*. Hal ini disebabkan karena masing-masing solusi ditelusuri secara node/graf, dan data berasal dari proses acak *local search* diproses secara probabilistik sehingga terdapat banyak constraint yang terlanggar.

Sedangkan pada *Tabu Search* perhitungan iterasi dan diambil Global min atau hasil terbaik pada proses iterasi, melalui data pencocokan pada *tabu list*,

sehingga dapat mengurangi banyaknya constraint yang terlanggar.

4.3.2 Durasi waktu

Perbandingan Tabu Search dan ACO dilihat dari durasi waktu yang dilakukan dapat dilihat pada gambar 5 berikut:

Gambar 5.Perbandingan Tabu Search dan ACO dari Global Min/Timeslot

Keterangan:

Dari grafik dapat dilihat bahwa hasil penjadwalan ACO memiliki nilai waktu lebih besar daripada Tabu Search . Hal ini disebabkan karena pada ACO masing-masing solusi ditelusuri secara node/graf, sehingga membutuhkan waktu yang lebih lama, sedangkan Tabu Search mengambil local solution untuk perhitungan iterasi dan diambil Global minnya saja, sehingga membutuhkan waktu relatif lebih cepat. Sedangkan pada ACO, penelusuran node/graf dilakukan untuk memastikan bahwa jadwal tidak akan bentrok dengan waktu dan ruangan, sehingga memakan waktu lebih lama. Metode *Tabu Search* dapat digunakan sebagai sistem penjadwalan kuliah, karena metode ini memilih langkah berikutnya (*neighbour- solution)* berdasarkan *constraint* dan penalti. Begitu pula ACO, memberikan solusi untuk masalah penjadwalan dengan memperhitungkan waktu meskipun variabel *local solution* dilakukan secara probabilistik.

Sulit untuk mendapatkan jadwal dengan tingkat kebenaran 100 %, karena tidak dapat menampilkan semua solusi yang terdapat dalam suatu kasus. Hal ini disebabkan karena Tabu Search tidak memeriksa semua *neighbour solution* yang ada, melainkan hanya memeriksa *neighbour solution* yang memiliki kemungkinan terbesar untuk memecahkan masalah, sehingga secara otomatis waktu yang diperlukan untuk suatu solusi yang memiliki tingkat kebenaran yang cukup tinggi akan menjadi lebih cepat, dibandingkan dengan metode ACO dimana aplikasi penjadwalan mata kuliah ini berbasis pada proses acak *local search* sehinga memakan waktu yang lebih lama dan dipercepat oleh algoritma semut. Hasil penjadwalan kuliah merupakan Global Min, atau hasil terbaik dari proses iterasi pada Tabu Search, sedangkan pada ACO hasil yang baik cenderung diperoleh pada iterasi-iterasi besar, hal ini membuktikan bahwa update *pheromone* berpengaruh terhadap penelusuran semut dalam menghasilkan solusi, dan proses ini dipengaruhi oleh jumlah komponen yang dijadwalkan, semakin besar kasus maka waktu yang diperlukan akan semakin lama pula.

Waktu yang digunakan untuk memperoleh jadwal dengan tingkat kebenaran 90 % jauh Tabu Search lebih cepat dibandingkan dengan metode ACO, karena pada ACO *variabel local solution* dilakukan secara probabilistik sedangkan pada Iterasi *Tabu Search* dihentikan saat semua langkah tidak dapat memperbaiki *current condition* dan semua *neighbour solution* sudah ada didalam *Tabu List.*

5. SIMPULAN DAN SARAN

5.1 Simpulan

Pada bab ini dijelaskan beberapa simpulan sesuai dengan uraian yang telah dijelaskan pada bagian-bagian sebelumnya dan saran bagi pengembangan terhadap perangkat lunak yang dibuat. Dengan memerhatikan program yang telah dibuat didapatkan beberapa simpulan, antara lain :

1. Aplikasi penjadwalan perkuliahan dengan menggunakan metode ACO dan Tabu Search didesain dengan menggunakan pemrograman Microsoft Visual Basic dengan memasukkan beberapa parameter yaitu Data Dosen, Mata Kuliah, Ruangan, dan beberapa variabel dari masing-masing parameter yang kemudian diproses sehingga menghasilkan penjadwalan perkuliahan.
2. Proses penjadwalan dengan Metode ACO pada *software* dilakukan dengan memasukkan data Dosen, Mata Kuliah dan Ruangan sebagai proses acak *local search* yang kemudian masuk dalam proses penjadwalan dan menghasilkan output berupa jadwal kuliah. Sedangkan pada *Tabu Search*, parameter Data Mata Kuliah, Dosen, dan Ruangan merupakan *local search* yang dimasukkan dalam proses pencocokan (*tabu list*) terlebih dahulu yang selanjutnya menghasilkan output jadwal mata kuliah.
3. Hasil penelitian untuk membandingkan unjuk kerja antara kedua metode dilihat dari jumlah *constrain* yang terlanggar serta lamanya waktu yang diperlukan dari masing-masing iterasi pada tiap metode sampai mendapatkan jadwal kuliah, dimana hasilnya:

   a. Pelanggaran constrain terbesar pada metode ACO yaitu 14 sedangkan untuk Metodre *Tabu Search* terjadi pelanggaran sebesar 8.

   b. Waktu yang digunakan untuk memperoleh jadwal dengan tingkat kebenaran 90 % yaitu t*abu search* dimana prosesnya lebih cepat, dan hasil terbaik dari proses iterasi (*Global min*) dibandingkan dengan metode ACO karena berasal dari proses acak *local search.*

5.2 Saran

Dalam hal ini penulis menyadari bahwa sistem ini masih banyak kekurangan dan kelemahan oleh karena itu disarankan untuk :

1. Dapat mengembangkan aplikasi ini dengan bahasa pemrograman yang lain, dan metode lain agar dapat menampilkan semua solusi yang terdapat dalam sebuah problem penjadwalan kuliah atau problem sejenisnya.

Dapat mengembangkan aplikasi ini dengan memperbaiki kekurangan yang dimiliki oleh aplikasi ini. Sehingga dapat menghasilkan aplikasi yang lebih baik.

2. Untuk pengembangan yang mungkin, disarankan analisa algoritma *Tabu Search* dan ACO diterapkan dalam sistem yang lebih kompleks dengan memperhatikan daerah permasalahan yang ditangani sebagai berikut:

   a. Menerapkan algoritma *Tabu Search* dan ACO dalam simulasi sistem yang lain bukan penjadwalan mata pelajaran sehingga didapat hasil yang optimal.

   b. Jika menggunakan simulasi sistem penjadwalan mata pelajaran, komponen yang termasuk ke dalam *Hard Constraint* dijadikan prioritas utama, sehingga sistem dalam melakukan pencarian yang benar-benar efektif.

   c. Untuk pengembangan yang lebih lanjut penggunaan algoritma *Tabu* Search dan ACO untuk penjadwalan penggunaanya harus dijalankan pada perangkat komputer yang dilengkapi dengan processor berkecepatan tinggi mengingat algoritma *Tabu Search* menerapkan konsep *adaptive memory*.

DAFTAR PUSTAKA

[1] Colorni, M. Dorigo et V. Maniezzo, *Distributed Optimization by Ant Colonies*, actes de la première conférence européenne sur la vie artificielle, Paris, France, Elsevier Publishing, 134-142, 1991.

[2] M. Dorigo, V. Maniezzo, et A. Colorni, *Ant system: optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man, and Cybernetics--Part B , volume 26, numéro 1, pages 29-41, 1996.

[3] T. Stützle et H.H. Hoos, *MAX MIN Ant System*, Future Generation Computer Systems, volume 16, pages 889-914, 2000

[4] Kusumadewi Sri, *Artificial Intelligence (Teknik dan Aplikasinya)* Yogyakarta : Graha Ilmu, 2003.

[5] Kusumadewi Sri dan Purnomo Hari, P*enyelesaian Masalah Optimasi dengan Teknik-teknik Heuristik*, Yogyakarta : Graha Ilmu, 2005.

[6] D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck, B. Baesens, *Classification with Ant Colony Optimization*, IEEE Transactions on Evolutionary Computation, volume 11, number 5, pages 651—665, 2007.

[7] Nugroho, Adi Paulus, Desain dan Implementasi Business Intelligence Untuk Mendukung Pembuatan Keputusan, Tesis Magister, Institut Teknologi Bandung, 2003.

[8] Favaretto, Daniel Elena Moretti, Paola Pellegrini. (2007). Ant colony system for a VRP with multiple time windows and multiple visits. Department of Applied Mathematics. University Ca' Foscari of Venice Dorsoduro 3825/EI-30123 Venice, Italy

[9] Pablo Ortega1 Cristian Oliva2 Jacques Ferland3 Manuel Cepeda2. (2009). Multiple Ant Colony System For A Vrp With Time Windows And Scheduled LoadinG. Ingeniare. Revista chilena de ingenieria, vol. 17 No 3, 2009, pp. 393-403