RANCANG BANGUN APLIKASI PENJADWALAN MATA KULIAH MENGGUNAKAN PARTICLE SWARM OPTIMIZATION (PSO)

(Studi Kasus Jurusan Teknik Informatika UIN Suska Riau)

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Pada
Jurusan Teknik Informatika

oleh :

IRFRANS KUSMARNA
10651004378

UIN SUSKA RIAU

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU
PEKANBARU
2013

RANCANG BANGUN APLIKASI PENJADWALAN MATA KULIAH MENGGUNAKAN *PARTICLE SWARM OPTIMIZATION* (PSO)

(Studi Kasus Jurusan Teknik Informatika UIN Suska Riau)

IRFRANS KUSMARNA

NIM : 10651004378

Tanggal Sidang : 05 Juni 2013

Tanggal Wisuda : Nopember 2013

Jurusan Teknik Informatika

Fakultas Sains dan Teknologi

Universitas Islam Negeri Sultan Syarif Kasim Riau

Jl. Soebrantas KM 15 No. 155 Pekanbaru

ABSTRAK

Penjadwalan mata kuliah merupakan hal yang penting di dunia pendidikan, karena semua kegiatan belajar mengajar bergantung pada jadwal yang disediakan, sehingga harus disusun dengan benar dan diperbaiki pada awal tahun akademik, sehingga nantinya tidak mengganggu aktifitas belajar mengajar. Untuk menyelesaikan masalah tersebut digunakan algoritma *Particle Swarm Optimization* (PSO). Algoritma ini dapat memecahkan masalah dengan membentuk partikel-partikel pada populasi awal secara acak, mengevaluasi nilai *fitness*, dan meng-*update velocity* serta posisi dari partikel. Ini bertujuan memecahkan permasalahan yang ditinjau dari fungsi *fitness* setiap partikel. Dari hasil pengujian, aplikasi penjadwalan perkuliahan meggunakan algoritma PSO mampu menghasilkan jadwal perkuliahan yang sudah tidak terjadi bentrokan walaupun masih tidak memenuhi dari segi kualitas yaitu jam dimulainya perkuliahan difokuskan pada jam-jam yang efektif.

Kata Kunci : *Constraint, Fitness, Particle swarm optimizaton,* Penjadwalan perkuliahan

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

Assalamu'alaikum Warahmatullahi Wabarakatuh,

Puji syukur kehadirat Allah Subhanahu wa Ta'ala, yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan laporan Tugas Akhir ini. Tidak lupa shalawat dan salam kepada junjungan Nabi Muhammad Shalallahu 'Alaihi Wasallam yang merupakan suri tauladan bagi umat manusia.

Penulis menyadari bahwa selesainya laporan Tugas Akhir dengan judul **RANCANG BANGUN APLIKASI PENJADWALAN MATA KULIAH MENGGUNAKAN *PARTICLE SWARM OPTIMIZATION* (PSO)** tidak terlepas dari dukungan berbagai pihak. Oleh karena itu, pada kesempatan ini penulis mengucapkan terimakasih sebesar-besarnya kepada :

1. Ayahanda dan Ibunda tercinta yang selalu memberikan doa, motivasi, bimbingan yang tiada hentinya, serta telah banyak berkorban demi keberhasilan anak-anaknya. Semoga mereka selalu dalam lindungan Allah Subhanahu Wa Ta'ala dan segala pengorbanan yang mereka berikan mendapat pahala dari Allah Subhanahu Wa Ta'ala, Aamiin.
2. Kakanda Irlov Kusmarna, AMK, Irjen Kusmarna, SE, dan Irzet Kusmarna, S.Kep, yang selalu memberikan dukungan dan semangatnya, serta dua keponakan tersayang, Zaskia Nia Lovina dan Rayyan Izza Lovino, yang selalu menjadi penyemangat dan penghilang lelah untuk menyelesaikan tugas akhir ini.
3. Bapak Prof. DR. H. M. Nazir, selaku Rektor Universitas Islam Negeri Sultan Syarif Kasim Riau.
4. Ibu Dra. Hj. Yenita Morena, M.Si, selaku Dekan Fakultas Sains dan Teknologi, Universitas Islam Negeri Sultan Syarif Kasim Riau.
5. Bapak Dr. Okfalisa, ST, M.Sc., selaku Ketua Jurusan Teknik Informatika.
6. Ibu Luh Kesuma Wardhani, ST, MT selaku pembimbing I Tugas Akhir.
7. Bapak Muhammad Safrizal, ST, M.Cs selaku pembimbing II Tugas Akhir.

8. Bapak Benny Sukma Negara, ST, MT selaku Penguji I Tugas Akhir.
9. Ibu Fitri Wulandari, S.Si, M.Kom selaku Penguji II Tugas Akhir.
10. Ibu Lestari Handayani, ST, M.Kom atas bantuan, pemikiran, dan kritikan-kritikan yang membangun dalam penyelesaian tugas akhir ini.
11. Pimpinan, seluruh Dosen, staf dan karyawan Jurusan Teknik Informatika.
12. Ibu Dewi Anggriani Harahap, M.Keb, Puket I STIKes Tuanku Tambusai Riau, atas bantuan, pemikiran, dan kritikan-kritikan yang membangun dalam penyelesaian tugas akhir ini.
13. Sahabat terbaikku, Yudi Nasrendra, ST dan Alfi Syahri, ST, terima kasih untuk bantuan yang diberikan selama ini.
14. Seluruh teman-teman seperjuangan angkatan 2006, khususnya TIF C.
15. Dan semua pihak yang telah membantu yang tidak bisa disebutkan satu persatu.

Penulis menyadari bahwa tugas akhir ini masih jauh dari sempurna. Oleh karena itu, kritik serta saran yang membangun dari rekan-rekan pembaca sangat dibutuhkan agar dapat membuat tugas akhir ini lebih baik. Akhir kata penulis berharap agar tugas akhir ini bisa memberikan manfaat bagi pembaca dan semua pihak yang berkepentingan. Terima kasih.

Pekanbaru, 05 Juni 2013

IRFRANS KUSMARNA

DAFTAR ISI

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perguruan Tinggi merupakan institusi yang memiliki peran dan posisi strategis dalam pencapaian tujuan pendidikan, yang mana hal ini memerlukan upaya perbaikan yang harus dilakukan secara terus-menerus untuk mewujudkan Sumber Daya Manusia (SDM) yang berkualitas. SDM merupakan sumber daya yang penting di suatu perguruan tinggi, karena tanpa adanya unsur manusia dalam perguruan tinggi, tidak mungkin perguruan tinggi tersebut mampu bergerak dan mencapai tujuan yang telah ditetapkan.

Penjadwalan merupakan proses, cara, pembagian waktu berdasarkan rencana pengaturan yang terperinci. Terdapat banyak hal yang harus dijadwalkan di suatu perguruan tinggi, diantaranya proses penerimaan mahasiswa baru, rapat intern universitas, seminar, Ujian Tengah Semester (UTS), Ujian Akhir Semester (UAS), dan yang paling penting yaitu sebagai motorik dari universitas yaitu penjadwalan mata kuliah. Penjadwalan mata kuliah merupakan hal yang penting di dunia pendidikan. Pada proses inilah dosen dan mahasiswa dapat saling berbagi ilmu pengetahuan di waktu dan ruangan yang tepat.

Universitas Islam Negeri Sultan Syarif Kasim (UIN Suska) Riau merupakan perguruan tinggi negeri yang terdiri dari beberapa fakultas, salah satunya adalah Fakultas Sains dan Teknologi. Fakultas Sains dan Teknologi juga memiliki beberapa jurusan, yang salah satunya adalah Jurusan Teknik Informatika. Berdasarkan hasil wawancara yang dilakukan oleh calon peneliti dengan Sekretaris Jurusan Teknik Informatika UIN Suska Riau, terdapat beberapa aturan yang harus diperhatikan pada saat penjadwalan mata kuliah di jurusan Teknik Informatika UIN Suska, yaitu: a) setiap mata kuliah disajikan maksimal 2 (dua) kali sehari; b) setiap dosen dijadwalkan mengajar 2 (dua) kali sehari; c) tidak terdapat perkuliahan pada jam shalat jumat; d) tidak terdapat perkuliahan pada jam makan siang; e) mata kuliah inti dijadwalkan pagi hari; f) dosen praktisi dijadwalkan mengajar hari sabtu; g) dosen TIF dan non-TIF dijadwalkan

mengajar pada hari senin-jumat. h) Mata kuliah pilihan dijadwalkan siang hari; i) Tidak ada perkuliahan pada hari Kamis dari pukul 08.00–12.00 bagi mahasiswa semester I, II, dan III; j) Waktu perkuliahan yang tersedia adalah hari Senin–Rabu adalah antara pukul 08.00–16.00 WIB, dan hari Kamis–Jumat antara pukul 08.00–16.30 WIB.

Untuk membuat jadwal mata kuliah yang baik, harus memperhatikan berbagai aspek, yaitu dari aspek dosen, mahasiswa dan ruang kelas. Pada penelitian ini yang akan menjadi sentral penelitian adalah penjadwalan dari aspek dosen dan pemakaian ruang kelas, karena untuk membagi dosen sesuai dengan bidang mata kuliahnya dan waktu yang cocok diperlukan pengaturan yang cukup rumit serta jumlah ruang kelas yang bisa dipakai untuk perkuliahan terbatas sehingga perlu pengoptimalan penggunaan ruangan.

Penelitian tentang penjadwalan mata kuliah ini sebelumnya juga pernah dirancang dan diaplikasikan oleh Heni Rachmawati dari Institut Teknologi Sepuluh Nopember (ITS), Rudiyanto dari Universitas Komputer Indonesia, dan Yendrika Putra dari Jurusan Teknik Informatika UIN Suska Riau dengan menggunakan metode Pewarnaan *Graph* dengan Algoritma Koloni Lebah, Algoritma *Max-Min Ant System*, dan Algoritma Genetika. Pada metode pewarnaan *graph* dengan Algoritma Koloni Lebah yang disusun oleh Heni Rachmawati (Rachmawati, 2012) menyimpulkan bahwa penggunaan algoritma koloni lebah mampu menyelesaikan pewarnaan *graph* dengan baik. Pada metode *Max-Min Ant System* yang disusun oleh Rusdiyanto (Rusdiyanto, 2006), dapat menyelesaikan permasalahan penjadwalan perkuliahan dengan baik. Sedangkan pada metode Algoritma Genetika yang disusun oleh Yendrika Putra (Putra, 2009) sudah berhasil memenuhi semua *constraint* yang ditetapkan.

Salah satu metode yang dapat digunakan untuk menyelesaikan permasalahan tersebut adalah dengan menggunakan metode *Particle Swarm Optimization* (PSO). *Particle Swarm Optimization* (PSO) merupakan bagian dari pencarian *heuristic*. *Particle Swarm Optimization* (PSO) adalah suatu metode optimasi yang menggabungkan metode *local search* dengan metode *global search*, yang menyeimbangkan antara eksplorasi dan eksploitasi. Diharapkan

dengan adanya *Particle Swarm Optimization* (PSO) akan diperoleh optimasi penjadwalan yaitu kondisi di mana terjadi kombinasi terbaik untuk pasangan mata kuliah dan dosen secara keseluruhan, tidak ada persolaan bentrokan jadwal pada sisi mahasiswa.

Penelitian mengenai penjadwalan mata kuliah menggunakan PSO ini pernah diteliti oleh Ariani dari Jurusan Teknik Informatika Politeknik Elektronika Negeri Surabaya (PENS) yang dalam salah satu kesimpulannya mengatakan bahwa penjadwalan mata kuliah dengan metode PSO ini menghasilkan jadwal yang optimal tanpa pelanggaran konstrain, yaitu sudah tidak ada jadwal mengajar dosen yang bentrok, sudah tidak ada mahasiswa yang kuliah lebih dari satu mata kuliah pada hari dan jam yang sama, sudah tidak ada dosen yg mengajar mata kuliah yang sama pada satu hari, dan sudah tidak ada mata kuliah yang dijadwalkan menempati ruang kelas atau laboratorium yang sama pada hari dan jam yang sama.

Selain berdasarkan bentrok atau tidak bentroknya jadwal mengajar dosen, dan ruangan perkuliahan, yang membedakan penelitian Dian Ariani terhadap penelitian yang akan dibuat adalah adanya tambahan *constraint* yaitu tidak adanya perkuliahan pada jam shalat Jumat dan tidak adanya perkuliahan pada jam istirahat.

Berdasarkan uraian di atas perlu dibuat sebuah aplikasi yang menggunakan metode yang tepat untuk membantu penyusunan jadwal perkuliahan di Jurusan Teknik Informatika UIN Suska Riau. Salah satunya dengan menggunakan metode *Particle Swarm Optimization* (PSO). Diharapkan dengan bantuan *Particle Swarm Optimization* (PSO) penyusunan penjadwalan mata kuliah dapat dioptimalkan.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dari Tugas Akhir ini adalah, “Bagaimana merancang dan membangun aplikasi penjadwalan dengan menggunakan *Particle Swarm Optimization* (PSO) pada Jurusan Teknik Informatika UIN Suska Riau”.

1.3 Batasan Masalah

Batasan dalam Tugas Akhir ini disesuaikan dengan keadaaan yang terdapat pada Jurusan Teknik Informatika UIN Suska Riau, yaitu:

1. Mata kuliah yang akan mengalami proses dalam *Particle Swarm Optimization* (PSO) hanya mata kuliah yang sifatnya memerlukan ruangan, hari dan waktu tertentu.
2. *Constraint* yang digunakan bersifat statis yang diinputkan pada program, diantaranya tidak terdapat bentrok jadwal mengajar dosen, tidak adanya bentrok pemakaian ruangan, tidak adanya perkuliahan pada jam shalat Jumat.
3. Data uji coba yang digunakan adalah data mata kuliah semester genap TA. 2011/2012.

1.4 Tujuan Penelitian

Tujuan yang ingin dicapai dari penulisan Tugas Akhir ini adalah sebagai berikut:

1. Menyusun jadwal perkuliahan yang otomatis.
2. Menerapkan metode PSO untuk menyusun jadwal perkuliahan.
3. Membuat aplikasi penjadwalan mata kuliah yang menerapkan metode PSO.

1.5 Sistematika Penulisan

Laporan tugas akhir ini terdiri dari enam bab, dengan sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN

Membahas mengenai latar belakang permasalahan, rumusan masalah, batasan masalah, tujuan pembahasan, metodologi penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Membahas teori-teori pendukung. Teori yang diangkat yaitu mengenai sistem pendidikan di Jurusan Teknik Informatika, penjadwalan, *Particle Swarm Optimization*

BAB III METODOLOGI PENELITIAN

Membahas tahapan penelitian, yaitu pengumpulan data, studi pustaka, analisa kebutuhan, perumusan masalah, analisa PSO, analsisa sistem, perancangan perangkat lunak, implementasi, pengujian sistem, dan kesimpulan akhir.

BAB IV ANALISIS DAN PERANCANGAN

Membahas tentang analisa sistem lama, sistem baru yang akan dikembangkan dan rancangan sistem penjadwalan mata kuliah dengan menggunakan metode *Particle Swarm Optimization* (PSO).

BAB V IMPLEMENTASI DAN PENGUJIAN

Menjelaskan implementasi sistem yang meliputi lingkungan implementasi aplikasi penjadwalan mata kuliah dan pengujian sistem yang meliputi lingkungan pengujian dan hasil pengujian.

BAB VI PENUTUP

Bab ini berisikan kesimpulan dari tugas akhir yang dibuat dan menjelaskan saran-saran penulis kepada pembaca agar penerapan metode *Particle Swarm Optimization* (PSO) dapat dikembangkan lagi.

BAB II
TINJAUAN PUSTAKA

2.1 Penjadwalan

Berdasarkan Kamus Besar Bahasa Indonesia, jadwal merupakan pembagian waktu berdasarkan rencana pengaturan urutan kerja. Jadwal juga didefinisikan sebagai daftar atau tabel kegiatan atau rencana kegiatan dengan pembagian waktu pelaksanaan yang terperinci. Sedangkan penjadwalan merupakan proses, cara, perbuatan menjadwalkan atau memasukkan dalam jadwal (Depdikbud, 1995). Sedangkan menurut Putra (2009), penjadwalan merupakan proses untuk menyusun suatu jadwal atau urutan proses yang diperlukan dalam sebuah persoalan. Persoalan penjadwalan biasanya berhubungan dengan penjadwalan kelas dalam sekolah atau perkuliahan dan juga dalam lingkup yang tidak jauh berbeda seperti penjadwalan ujian, penjadwalan karyawan, ataupun penjadwalan *job shop*. Dalam penjadwalan kuliah, akan dibahas tentang pembagian jadwal untuk tiap mahasiswa pada kuliah tertentu sekaligus dosen pengajarnya. Dalam penjadwalan ujian akan dibahas pengaturan dosen yang menjaga ujian dan mahasiswa yang menempati ruang ujian yang ada. Dalam penjadwalan karyawan, dilakukan pengaturan karyawan yang akan bekerja pada waktu tertentu di bagian tertentu. Sedangkan dalam penjadwalan *job shop*, dilakukan penjadwalan sejumlah mesin dan sejumlah pekerjaan terkait rute yang telah ditentukan.

Proses tersebut tentu saja dibuat berdasarkan permasalahan yang ada. Beberapa proses umum yang perlu dilakukan untuk menyelesaikan suatu proses penjadwalan sebagai berikut (Putra, 2009):

a. Mendefinisikan atau membuat model dari permasalahan. Model yang dibuat mencakup proses apa saja yang akan dikerjakan dalam persoalan penjadwalan yang ada. Atau lebih jelasnya jadwal apa saja yang akan dibuat;

b. Mendesain metode penyelesaian untuk permasalahan penjadwalan tersebut. Dari model yang telah ada, ditentukan metode yang akan digunakan untuk menyelesaikan permasalahan penjadwalan tersebut;
c. Mencari bermacam-macam contoh permasalahan penjadwalan yang telah dibuat. Dalam proses ini dilakukan pencarian penyelesaian penjadwalan yang pernah digunakan agar dapat dipakai sebagai referensi dalam proses yang sedang dilakukan.

2.2 Sistem Pendidikan di Jurusan Teknik Informatika UIN Suska

Sistem pendidikan yang dipakai oleh Jurusan Teknik Informatika UIN Suska mengikuti sistem pendidikan yang dirancang oleh UIN Suska, yaitu sistem Satuan Kredit Semester (SKS). Sistem ini diberlakukan sebagai implikasi dari pola pendidikan tinggi di Indonesia.

2.2.1 Pengertian SKS

a. Kredit adalah satuan yang digunakan untuk menyatakan beban studi mahasiswa, beban mengajar dosen dalam satu semester dan pengakuan atas keberhasilan mahasiswa.
b. Semester adalah satuan terkecil untuk menyatakan lama satu program pendidikan dalam jenjang pendidikan, satu semester setara dengan 16 minggu perkuliahan termasuk evaluasi, atau sama dengan 16 - 18 minggu kerja.
c. Satuan Kredit Semester (SKS) adalah satuan untuk menyatakan besarnya program pendidikan dalam satu semester.
    1. Satu SKS setara dengan (UIN Suska, 2006):
        - 50 menit kegiatan tatap muka berjadwal antara mahasiswa dan tenaga pengajar, misalnya dalam bentuk kuliah dan diskusi;
        - 60 menit kegiatan akademik terstruktur;
        - 60 menit kegiatan mahasiswa mandiri;
    2. Satu SKS untuk praktikum setara dengan :
        - 50 menit praktikum terjadwal di laboratorium;
        - 60 menit kegiatan akademik terstruktur;

– 60 menit kegiatan akademik mandiri;

2.2.2 Ciri-Ciri Utama SKS

Sistem Satuan Kredit Semester (SKS) memiliki ciri-ciri sebagai berikut (UIN, 2006):

1. Kepada mahasiswa ditawarkan program pendidikan yang bervariasi;
2. Mahasiswa menyusun kombinasi program yang akan diikuti sesuai dengan minat dan bakatnya;
3. Memungkinkan pindahnya dari satu program ke program lainnya tanpa kehilangan tabungan kredit semester yang pernah diperolehnya. Tabungan semester yang telah diperolehnya dapat ditransfer ke program pendidikan yang baru;
4. Menggunakan sarana lebih efisien.

2.3 *Artificial Intelligence* (**Kecerdasan Buatan**)

Artificial Intelligence (AI) atau Kecerdasan Buatan merupakan salah satu bagian ilmu komputer yang membuat agar mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia (Kusumadewi, 2003). Menurut Rich (1991) AI adalah sebuah studi tentang bagaimana membuat komputer mengerjakan sesuatu yang dapat dikerjakan manusia. Menurut Setiawan (1993), AI adalah cabang ilmu komputer yang mempelajari otomatisasi tingkah laku cerdas. Menurut Turing, dkk., (1996), AI didefinisikan suatu perilaku sebuah mesin yang jika dikerjakan oleh manusia akan disebut cerdas (Ahmad, 2006).

Menurut Stuart Russel dan Peter Norvig (1995), mengelompokkan definisi AI ke dalam empat kategori, yaitu:

1. *Thinking Humanly : The Cognitive Modelling Approach*

   Pendekatan ini dilakukan dengan dua cara, yaitu:

   - Melalui introspeksi : mencoba menangkap pemikiran-pemikiran kita sendiri pada saat kita berfikir;
   - Melalui eksperimen-eksperimen psikologi.
2. *Acting Humanly : the Turing Test Approach*
3. *Thinking rationally : the Laws Thought Approach*

4. *Acting rationally : the Rational Agent Approach* (*Stuart Russel dan Peter Norvig, dikutip oleh Suyanto, 2007*).

Definisi AI yang paling tepat untuk saat ini adalah *action rationally* dengan pendekatan *rational agent*. Hal ini berdasarkan pemikiran bahwa komputer bisa melakukan penalaran secara logis dan juga bisa melakukan aksi secara rasional berdasarkan hasil penalaran tersebut (Suyanto, 2007).

Sejak pertama kali dikemukakan istilah AI pada tahun 1956 di konferensi Darthmouth, AI terus dikembangkan melalui berbagai penelitian mengenai teori-teori dan prinsip-prinsipnya. Perkembangan AI mengalami pasang surut mengikuti antusias para peneliti dan dana penelitian yang tersedia. Pada periode 1966 sampai 1974, perkembangan AI melambat. Tetapi sejak tahun 1980, AI menjadi sebuah industri yang besar dengan dengan perkembangan yang sangat pesat. Banyak industri skala besar yang melakukan investasi besar-besaran dalam bidang AI.

Saat ini, dengan semakin cepatnya perkembangan *hardware* dan *software*, berbagai produk AI telah berhasil dibangun dan digunakan dalam kehidupan sehari-hari. Dengan teknologi *hardware* yang performansinya semakin tinggi dan berukuran kecil serta didukung teknologi *software* yang semakin beragam dan kuat, produk-produk berbasis AI semakin dekat dengan kehidupan manusia.

Pada masa mendatang, AI ditantang untuk membuat suatu kecerdasan yang hampir menyamai kecerdasan manusia. Ray Kurzweil memprediksi bahwa hal itu akan mungkin terwujud sekitar 90 tahun ke depan (tahun 2099) (Suyanto, 2007).

2.4 Pencarian *Heuristic*

Pencarian atau *searching* adalah teknik penyelesaian masalah yang merepresentasikan masalah ke dalam ruang keadaan (*state*) dan secara sistematis melakukan pembangkitan dan pengujian *state-state* dari *initial state* sampai ditemukan suatu *goal state*.

Heuristik (*heuristic*) berasal dari sebuah kata kerja bahasa Yunani, *Heuriskein*, yang berarti “mencari” atau “menemukan”. Dalam dunia

pemrograman, sebagian orang menggunakan kata heuristik sebagai lawan kata dari algoritmik, di mana kata heuristik ini diartikan sebagai “suatu proses yang mungkin dapat menyelesaikan suatu masalah, tetapi tidak ada jaminan bahwa solusi yang dicari selalu dapat ditemukan.” Dalam mempelajari metode-metode pencarian ini, kata heuristik diartikan sebagai suatu fungsi yang memberikan suatu nilai berupa biaya perkiraan (estimasi) dari suatu solusi (Suyanto, 2007).

Menurut Talbi (2009), metaheuristik dapat didefinisikan sebagai metode lanjut (*advanced*) berbasis heuristik untuk menyelesaikan persoalan optimasi secara efisien. Di dalam wikipedia, metaheuristik didefinisikan sebagai metode optimasi 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 optimasi stokastik (optimasi stokastik merupakan optimasi yang memiliki derajat ketidakpastian atau *random*).

Menurut Blum dan Roli (2003), metaheuristik memiliki beberapa karakteristik dasar, yaitu:

a. Metaheuristik adalah strategi yang memandu proses pencarian;
b. Tujuan dari metaheuristik adalah untuk menjelajahi ruang pencarian secara efisien untuk menemukan solusi optimal;
c. Teknik metaheuristik berkisar dari prosedur pencarian lokal yang sederhana sampai proses pembelajaran yang kompleks;
d. Metaheuristik adalah metode pendekatan dan biasanya non-deterministik;
e. Metaheuristik dapat terdiri dari penggabungan beberapa mekanisme supaya proses pencarian tidak terjebak dalam daerah terbatas di ruang pencarian;
f. Konsep dasar dari metaheuristik memungkinkan pendeskripsian secara abstrak;
g. Metaheuristik bersifat umum/*general* sehingga dapat diterapkan dalam berbagai macam persoalan;

h. Metaheuristik dapat memungkinkan pendeskripsian secara abstrak;
i. Metaheuristik dapat menggunakan pengalaman yang didapat selama proses pencarian untuk menuntun proses pencarian.

Dalam menentukan apakah metaheuristik adalah metode yang sesuai untuk menyelesaikan suatu permasalahan, ada beberapa hal yang perlu diperhatikan misalnya kompleksitas permasalahan, ukuran input, struktur input dan waktu yang diperlukan untuk menyelesaikan masalah tersebut. Secara umum, metaheuristik dipakai untuk masalah-masalah yang kompleks dan tidak bisa diselesaikan dengan mudah secara analitik/eksak. Tidaklah terlalu bermanfaat menggunakan metaheuristik untuk persoalan yang dengan mudah dan cepat dapat diselesaikan secara eksak (penyelesaian eksak merupakan penyelesaian terbaik, tetapi seringkali metode ini tidak dapat diterapkan pada permasalahan optimasi, sehingga dipakailah metode pendekatan (Hindriyanto, 2011).

2.5 *Particle Swarm Optimization*

2.5.1 Pengertian *Particle Swarm Optimization*

Particle Swarm Optimization, disingkat sebagai PSO, merupakan algoritma berbasis populasi yang mengeksploitasi individu dalam pencarian. Dalam PSO populasi disebut *swarm* dan individu disebut *particle*. Tiap partikel berpindah dengan kecepatan yang diadaptasi dari daerah pencarian dan menyimpannya sebagai posisi terbaik yang pernah dicapai.

PSO didasarkan pada perilaku sosial sekawanan burung atau sekumpulan ikan. Algoritma PSO meniru perilaku sosial organisme ini. Perilaku sosial terdiri dari tindakan individu dan pengaruh dari individu-individu lain dalam suatu kelompok. Kata partikel menunjukkan, misalnya, seekor burung dalam kawanan burung. Setiap individu atau partikel berperilaku secara terdistribusi dengan cara menggunakan kecerdasannya (*intelligence*) sendiri dan juga dipengaruhi perilaku kelompok kolektifnya. Dengan demikian, jika satu partikel atau seekor burung menemukan jalan yang tepat atau pendek menuju ke sumber makanan, sisa kelompok yang lain juga akan dapat segera mengikuti jalan tersebut meskipun lokasi mereka jauh di kelompok tersebut.

Metode optimasi yang didasarkan pada *swarm intelligence* ini disebut algoritma *behaviorully inspired* sebagai alternatif dari algoritma genetika, yang sering disebut *evolution-based procedures*. Algoritma PSO ini awalnya diusulkan oleh J. Kennedy dan R. C. Eberhart. Dalam konteks optimasi multivariabel, kawanan diasumsikan mempunyai ukuran tertentu atau tetap dengan setiap partikel posisi awalnya terletak di suatu lokasi yang acak dalam ruang multidimensi. Setiap partikel bergerak dalam ruang / *space* tertentu dan mengingat posisi terbaik yang pernah dilalui atau ditemukan terhadap sumber makanan atau nilai fungsi objektif. Setiap partikel menyampaikan informasi atau posisi bagusnya kepada partikel yang lain dan menyesuaikan posisi dan kecepatan masing-masing berdasarkan informasi yang diterima mengenai posisi yang bagus tersebut. Sebagai contoh, misalnya, perilaku burung-burung dalam kawanan burung. Meskipun setiap burung mempunyai keterbatasan dalam hal kecerdasan, biasanya ia akan mengikuti kebiasaan (*rule*) seperti berikut:

1. Seekor burung tidak berada terlalu dekat dengan burung yang lain;
2. Burung tersebut akan mengarahkan terbangnya ke arah rata-rata keseluruhan burung;
3. Akan memposisikan diri dengan rata-rata posisi burung yang lain dengan menjaga, sehingga jarak antar burung dalam kawanan itu tidak terlalu jauh.

Dengan demikian perilaku kawanan burung akan didasarkan pada kombinasi dari 3 (tiga) faktor simpel berikut:

1. Kohesi – terbang bersama
2. Separasi – jangan terlalu dekat
3. Penyesuaian (*alignment*) – mengikuti arah bersama

Jadi PSO dikembangkan dengan berdasarkan pada model berikut:

1. Ketika seekor burung mendeteksi target atau makanan (atau bisa minimum atau maksimum suatu fungsi tujuan) secara cepat mengirim informasi kepada burung-burung yang lain dalam kawanan tertentu.
2. Burung yang lain akan mengikuti arah menuju ke makanan tetapi tidak secara langsung.

3. Ada komponen yang tergantung pada fikiran burung, yaitu memorinya tentang apa yang sudah dilewati pada waktu sebelumnya.

Model ini akan disimulasikan dalam ruang dengan dimensi tertentu dengan sejumlah iterasi sehingga di setiap iterasi, posisi partikel akan semakin mengarah ke target yang dituju (minimum atau maksimum fungsi). Ini dilakukan hingga maksimum iterasi dicapai atau bisa digunakan kriteria penghentian yang lain.

PSO memiliki banyak kesamaan dengan teknik-teknik *evolutionary computation* yang lain, seperti *Genetic Algorithms (GA), Evolutionary Strategies (EA), Evolutionary Programming*, dan sebagainya. PSO maupun GA dimulai dengan suatu populasi yang terdiri dari sejumlah individu (yang menyatakan solusi) yang dibangkitkan secara acak dan selanjutnya melakukan pencarian solusi optimum melalui perbaikan individu untuk sejumlah generasi tertentu. Tetapi berbeda dengan GA, PSO tidak menggunakan operator-operator evolusi seperti rekombinasi (*cross over)* dan mutasi. PSO memiliki memori untuk menyimpan solusi terbaik, sedangkan GA tidak punya. Setiap partikel pada PSO tidak pernah mati, sedangkan individu pada GA bisa mati dan digantikan dengan individu baru. Pada PSO, posisi dan kecepatan terbang partikel di-*update* pada setiap iterasi sehingga partikel tersebut bisa menghasilkan solusi baru yang lebih baik.

Pada PSO akan digunakan algoritma dasar sederhana dari PSO dengan varibel bernilai *integer* dan faktor inersia *w* statis, tahapan-tahapan secara jelas digambarkan pada diagram alir seperti yang terlihat pada Gambar 2.1.

Algoritma PSO terdiri dari tiga tahap, yaitu pembangkitan posisi serta kecepatan partikel, *update velocity* (*update* kecepatan), *update position* (*update* posisi). Pertama posisi x_k^i dan kecepatan v_k^i dari kumpulan partikel dibangkitkan secara *random* menggunakan batas atas (x_{max}) dan batas bawah (x_{min}) dari *design variable*, seperti yang ditunjukkan pada persamaan (2.1) dan (2.2)

$$x_0^i = x_{min} + rand(x_{max} - x_{min}) \quad \text{..... (2.1)}$$

$$v_0^i = x_{min} + rand(x_{max} - x_{min}) \quad \text{..... (2.2)}$$

Di mana:

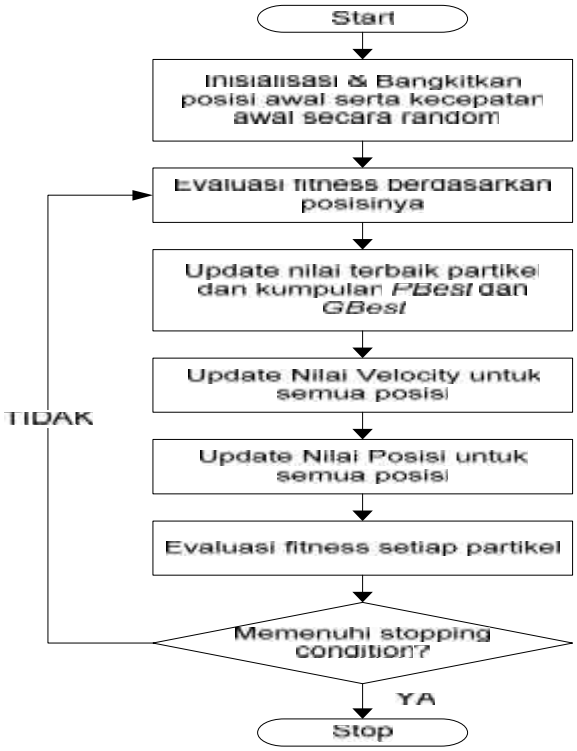
x_0^i = posisi awal

v_0^i = kecepatan awal

x_{min} = batas bawah

x_{max} = batas atas

rand = nilai *random* antara nilai 0 dan 1



Gambar 2.1 Diagram alir *PSO*

Posisi dan kecepatan direpresentasikan dalam bentuk vektor di mana *n* dimensi vektor merepresentasikan jumlah dari desain variabel partikel, dengan *superscript* dan *subscript* menotasikan partikel ke *i* pada waktu ke *k*. dengan proses inisialisasi ini maka kumpulan partikel dapat terdistribusi secara *random* pada *design space*. Vektor seperti ditunjukkan di bawah ini:

$x_k^i = (x_k^{i1}, x_k^{i2}, \ldots, x_k^{in})^T$

$v_k^i = (v_k^{i1}, v_k^{i2}, \ldots, v_k^{in})^T$

Langkah kedua adalah *update velocity* (kecepatan) untuk semua partikel pada waktu *k*+1 menggunakan fungsi objektif atau nilai *fitness* posisi partikel saat ini pada *design space* saat waktu ke *k*. Dari nilai *fitness* dapat ditentukan partikel mana yang memiliki nilai *global* terbaik (*global best*) pada *swarm* saat ini (p_k^g), dan juga dapat ditentukan posisi terbaik dari tiap partikel pada semua waktu yang

sekarang dan sebelumnya (p^i). Perumusan *update velocity* menggunakan dua informasi tersebut untuk semua partikel pada kumpulan dengan pengaruh perpindahan yang sekarang (v_k^i), untuk memberikan arah pencarian (v_{k+1}^i) untuk generasi selanjutnya. Ilustrasi *update velocity* dan *update* posisi partikel pada PSO seperti yang terlihat pada Gambar 2.2.

Perumusan *update velocity* mencakup beberapa parameter *random* (*rand*), untuk mendapatkan cakupan yang baik pada *design space*, tiga parameter yang mempengaruhi arah pencarian, yaitu *inertia factor* (*w*), *self confidence* (c_1), *swarm confidence* (c_2) akan digabungkan dalam satu penyajian, seperti yang ditunjukkan persamaan berikut:

$$v_{k+1}^i = w * v_k^i + c_1 * rand * (p^i - x_k^i + c_2 * rand * (p_k^g x_k^i \quad \text{..... (2.3)}$$

Di mana:

w = *inertia factor,* digunakan untuk mengontrol pengaruh kecepatan sebelumnya dikecepatan sekarang, mempengaruhi *trade-off* kemampuan *exploration* (menjelajah) *local* dan *global* selama proses pencarian. Nilai *w* memiliki rentang 0,4 – 0,9

v_k^i = kecepatan sekarang

x_k^i = posisi sekarang

c_1, c_2 = *self confidence, swarm confidence,* merupakan *learning rates* untuk kemampuan individu (*cognitive*) dan pengaruh sosial (*group*). parameters c_1 dan c_2 menunjukkan bobot dari memori (position) sebuah partikel terhadap memori (posisi) dari kelompok (*swarm*). Nilai dari c_1 dan c_2 biasanya adalah 2 sehingga perkalian c_1r_1 dan c_2r_2 memastikan bahwa partikel-partikel akan mendekati target sekitar setengah selisihnya

r_1, r_2 = bilangan *random* yang memiliki interval 0 dan 1

p^i = *local best*, posisi terbaik dari tiap partikel pada semua waktu yang sekarang

p_k^g = nilai *global* terbaik (*global best*) pada *swarm* saat ini

Langkah terakhir dari setiap iterasi adalah *update* posisi tiap partikel dengan vektor *velocity*, seperti yang ditunjukkan pada persamaan berikut:

$$x_{k+1}^i = x_k^i + v_{k+1}^i \quad \text{..... (2.4)}$$

Di mana :

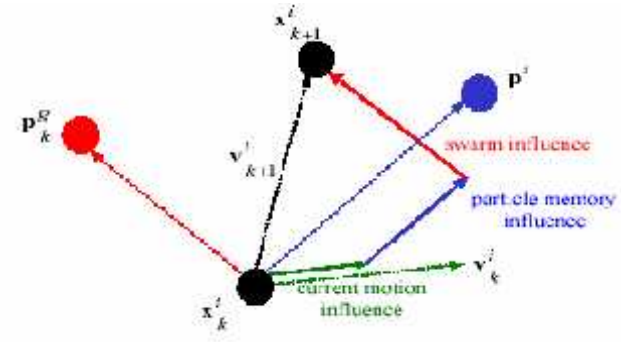
x_{k+1}^i = posisi pencarian

v_{k+1}^i = arah pencarian

x_k^i = posisi sekarang

Tiga tahapan di atas akan diulang sampai krtiteria kekonvergenan terpenuhi, kriteria kekonvergenan sangat penting dalam menghindari penambahan fungsi evaluasi setelah solusi optimum didapatkan, namun kriteria kekonvergenan tidak selalu mutlak diperlukan, penetapan jumlah iterasi maksimal juga dapat digunakan sebagai *stopping condition* dari algoritma.

Banyak cara untuk membangun kondisi berhenti, diantaranya adalah : iterasi dihentikan ketika PSO telah mencapai iterasi maksimum, atau PSO telah menemukan nilai optimum tertentu atau kesalahan minimum yang diinginkan.



Gambar 2.2 *Update velocity* dan posisi PSO

2.5.2 Algoritma *PSO*

Pseudo code algoritma PSO

```
for setiap partikel
     Inisialisasi partikel menggunakan persamaan (2.1) dan (2.2)
end
repeat
```

```
    for setiap partikel
        Hitung nilai fitness
            if nilai fitness baru lebih baik daripada
            nilai fitness lama
        Update nilai fitness dari partikel tersebut
    end
       Pilih partikel dengan nilai fitness terbaik diantara semua
       partikel tetangganya dan simpan nilai fitness terbaik
       tersebut
    for setiap partikel
        Hitung velocity partikel menggunakan
        persamaan (2.3)
        Update posisi partikel menggunakan persamaan (2.4)
    end
until (KriteriaBerhenti == true)
```

2.5.3 *Fitness*

Fungsi *fitness* digunakan untuk mengukur tingkat kebaikan atau kesesuaian (*fitness*) suatu solusi dengan solusi yang dicari. Fungsi *fitness* bisa berhubungan langsung dengan fungsi tujuan, atau bisa juga sedikit modifikasi terhadap fungsi tujuan. Sejumlah solusi yang dibangkitkan dalam populasi akan dievaluasi menggunakan fungsi *fitness*.

Pada kasus optimasi, dikenal dua masalah, yaitu maksimasi dan minimasi. Maksimasi artinya mencari nilai maksimal dari sesuatu (bisa berupa fungsi). Sedangkan minimasi artinya mencari nilai minimal dari sesuatu.

Jika tujuannya adalah memaksimalkan sebuah fungsi, maka fungsi *fitness* yang digunakan adalah fungsi itu sendiri, jadi $f = h$ (di mana f adalah fungsi *fitness*). Sedangkan jika tujuannya adalah meminimalkan fungsi h, maka fungsi h tidak bisa digunakan secara langsung. Fungsi *fitness* yang digunakan untuk masalah minimasi adalah $f = 1/h$. artinya, semakin kecil nilai h semakin besar nilai f. tetapi fungsi *fitness* ini akan bermasalah jika h bisa bernilai 0, yang mengakibatkan f bisa bernilai tak hingga. Untuk mengatasi masalah tersebut, fungsi *fitness* perlu dimodifikasi sedikit menjadi,

$$f = \frac{1}{(h+a)} \quad \text{..... (2.5)}$$

di mana *h* merupakan hasil penjumlahan dari *constraint* yang dilanggar. Sedangkan *a* adalah bilangan yang dianggap sangat kecil untuk menghindari pembagian dengan 0. Nilai *a* biasanya didefinisikan sebagai 0,001 atau disesuaikan dengan masalah yang akan diselesaikan.

2.5.4 Penjadwalan Mata Kuliah Dengan PSO

2.5.4.1 Pembangkitan Posisi dan *Velocity* Awal Partikel

Dalam penjadwalan mata kuliah ini posisi dan *velocity* dimisalkan sebagai slot. Posisi dan *velocity* awal ditentukan secara *random* dengan batasan nilai *minimum* slot dan *maximum* slot. Nilai posisi dan *velocity* awal partikel diinisialisasi sama, sehingga hanya melakukan sekali *random* untuk mendapatkan nilai *velocity* dan posisi partikel.

Pada penjadwalan ini posisi partikel diwakili oleh slot-slot, satu set jadwal yang terdiri dari beberapa kelas merupakan satu partikel. Setiap kelas memiliki beberapa slot dalam satu minggu perkuliahan, di mana setiap harinya terdapat 9 slot yang dapat digunakan dan setiap slot mewakili 1 jam kuliah.

Pembagian posisi dan *velocity* awal partikel dilakukan secara *random* dengan menggunakan persamaan (2.1) dan (2.2), di mana x_{min} (batas terkecil) adalah 0 dan x_{max} (batas terbesar) adalah *n*, kemudian mata kuliah dan dosen diletakkan pada posisi slot yang didapatkan secara *random*.

2.5.4.2 Menentukan Nilai *Fitness* Masing-Masing Partikel

Dalam penjadwalan ini nilai *fitness* ini menentukan banyaknya pelanggaran *constraint* yang harus dioptimasi.

Constraint-constraint yang digunakan untuk pengoptimasian aplikasi penjadwalan ini antara lain:

- Tidak boleh ada bentrok mahasiswa, yaitu setiap mahasiswa hanya diperbolehkan mengikuti satu matakuliah pada hari dan jam yang sama;
- Tidak boleh ada bentrok dosen, yaitu setiap dosen hanya diperbolehkan mengajar satu perkuliah pada hari dan jam yang sama;

– Tidak boleh ada dosen yang sama dan mengajar mata kuliah yang sama dalam satu kelas dalam sehari;
– Tidak boleh ada bentrok penggunaan lab.

Jika pada masing-masing partikel terjadi pelanggaran terhap *constraint-constraint* di atas, maka nilai *fitness*-nya adalah dijumlahkan setiap pelanggaran yang terjadi dalam satu iterasi kemudian dihitung dengan fungsi *fitness* minimasi. Sehingga partikel terbaik adalah partikel dengan nilai *fitness* terkecil.

2.5.4.3 Menentukan *Local Best* dan *Global Best*

Local Best (p^i)

Menentukan partikel yang terbaik dalam satu iterasi, yaitu partikel dengan nilai *fitness* paling kecil dari partikel-partikel lain dalam satu iterasi. Partikel terbaik tersebut kemudian disimpan sebagai *local best particle*.

Pseudo code mencari "*Local Best*"

```
For j = 1 To size
         If particlefitness(j) < f(pi(j)) Then
        For i = 1 To vektor
            pi(j, i) = particlevektor(j, i)
        Next i
    fpi(j)) = particlefitness(j)
End If
Next j
```

Global Best (p_k^g)

Menentukan partikel terbaik dari semua *particle best*/*local best*. Nilai *global best* pada iterasi pertama adalah sama dengan nilai *local best* pada iterasi pertama, kemudian untuk iterasi selanjutnya dilakukan *update*. Dan disimpan sebagai *global best particle*.

Pseudo code mencari '*global best*'

```
F(pg) = particlefitness(size)
For mem = 1 To size
    If (particlefitness(mem) < f(pg)) Then
       For i = 1 To vektor
          pg(i) = particlevektor(mem, i)
       Next i
    f(pg) = particlefitness(mem)
    End If
Next mem
```

2.5.4.4 Proses *Update Velocity* dan Posisi

Proses *update velocity* baru (v_{k+1}^i) ini menggunakan parameter nilai *velocity* yang lama (v_k^i), nilai *vector*/posisi yang lama (x_k^i), C1 (*learning rates local* partikel), C2 (*learning rates global* partikel), *local best* (p^i), *global best* (p_k^g) dan *random* bilangan acak dalam interval [0,1] dan untuk mendapatkan nilai *velocity* yang baru digunakan persamaan (2.3).

Pseudo code untuk *update velocity*

```
For j = 1 To size
    For i = 1 To vector
      Particlevelocity(j,i) = (particlevelocity(j,
      i)+(c1*rnd*(pi(j,i) – particlevector(j,i))) + (c2*rnd*(pg(i)
      – partclevector(j,i))))
   Next i
Next j
```

2.5.5 *Hard Constraint* dan *Soft Constraint*

Penjadwalan esensialnya merupakan *schedule* yang harus memenuhi sejumlah *constraint*. *Constraint* secara *universal* digunakan oleh individu yang berkaitan dengan permasalahan penjadwalan. *Constraint* dibagi menjadi dua kategori, yaitu *soft* dan *hard constraint* (Bambrick, 1997).

Hard constraint adalah *constraint* pada penjadwalan yang tidak boleh dilanggar. Sebagai contoh, dosen tidak boleh berada di tempat berlainan pada saat bersamaan (Bambrick, 1997). Beberapa contoh *hard constraint* diantaranya sebagai berikut:

1. Ruang kelas tidak boleh dipesan bersamaan;
2. Setiap perkuliahan harus dijadwalkan tepat satu kali;
3. Perkuliahan mahasiswa tidak boleh dijadwalkan secara bersamaan;
4. Dosen tidak boleh dijadwalkan mengajar dua waktu pada saat yang bersamaan;
5. Beberapa perkuliahan membutuhkan ruangan khusus;
6. Beberapa perkuliahan membutuhkan ruangan yang memiliki peralatan tertentu;
7. Beberapa perkuliahan harus berlangsung berkelanjutan;

8. Ruangan harus cukup besar untuk menampung setiap kelas yang dijadwalkan untuk itu;
9. Seorang dosen tidak harus dijadwalkan pada saat dia tidak bersedia. Misalnya, dosen yang memiliki perjanjian sebelumnya.

Soft constraint adalah *constraint* yang boleh dilanggar, tetapi pelanggaran itu harus diminimalkan. Sebagai contoh, perkuliahan harus dijadwalkan dekat dengan lokasi jurusan perkuliahan berlangsung. *Soft constraint* dalam beberapa penelitian sangat beragam, juga tingkat kepentingannya tergantung dari sumbernya (Bambrick, 1997). Contoh beberapa *soft constraint* adalah:

1. Beberapa dosen berharap perkuliahan dijadwalkan tidak secara berurutan pada waktu tertentu;
2. Ada jam tertentu untuk penjadwalan bagi perkuliahan dosen tertentu;
3. Banyak mahasiswa dan beberapa dosen berharap tidak ada periode waktu yang kosong dalam penjadwalan mereka;
4. Perkuliahan harus didistribusikan merata tiap minggu;
5. Ruangan kelas harus dijadwalkan dekat sesuai dengan prodi kelas tersebut;
6. Ruangan kelas yang dijadwalkan harus lebih besar daripada mahasiswa yang menghadiri perkuliahan.

2.6 Teknik Pengujian Perangkat Lunak

Pengujian perangkat lunak adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean (Pressman, 2000).

Teknik pengujian perangkat lunak dapat dibagi atas beberapa macam, namun yang paling umum digunakan adalah sebagai berikut:

1. Pengujian *White Box*

Pengujian *white box* terkadang juga disebut sebagai pangujian *glass-box*, merupakan metode perancangan *test case* yang menggunakan struktur *control* dari perancangan prosedural untuk memperoleh *test case.*

Dengan menggunakan metode *white-box,* sistem analis akan memperoleh *test case* yang :

a. Memberi jaminan bahwa jalur *independent* pada suatu modul telah digunakan paling tidak satu kali;
b. Menggunakan semua keputusan logis pada sisi *true* dan *false*;
c. Mengerjakan seluruh *loop* yang sesuai dengan batasannya;
d. Menggunakan struktur data *internal* yang menjamin validitas.

2. Pengujian *Black Box*

Pengujian *Black Box* adalah pengujian yang fokus pada persyaratan fungsional perangkat lunak. Pengujian ini memungkinkan sistem analis memperoleh kumpulan kondisi *input* yang akan mengerjakan seluruh keperluan fungsional program.

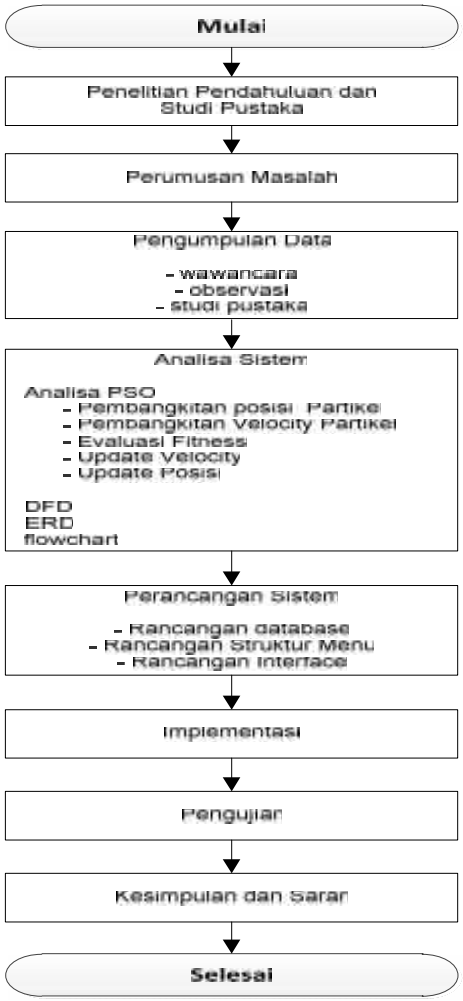
Tujuan metode ini mencari kesalahan pada:

a. Fungsi yang tidak benar atau hilang;
b. Kesalahan pada *interface*;
c. Kesalahan pada struktur data atau akses *database*;
d. Kesalahan kinerja;
e. Kesalahan inisialisasi dan tujuan akhir.

BAB III
METODOLOGI PENELITIAN

Metodologi penelitian adalah cara yang digunakan dalam memperoleh berbagai data untuk diproses menjadi informasi yang lebih akurat sesuai permasalahan yang akan diteliti. Metodologi penelitian dengan mendeskripsikan masalah yang dilengkapi dengan penyajian diagram alur pelaksanaan penelitian untuk memudahkan pemahaman tahapan penelitian.

Metodologi yang digunakan dalam penelitian tugas akhir yang berjudul ”Rancang Bangun Aplikasi Penjadwalan Mata Kuliah Menggunakan *Particle Swarm Optimization*” dapat di lihat pada Gambar 3.1. berikut ini.



Gambar 3.1. *Flowchart* metodologi penelitian

3.1 Penelitian Pendahuluan dan Studi Pustaka

Melakukan penelitian tahap awal untuk mencari informasi-informasi awal mengenai permasalahan yang dihadapi oleh pihak jurusan Teknik Informatika UIN Suska tentang penjadwalan mata kuliah, serta penelitian-penelitian terdahulu, yang berhubungan dengan pembuatan aplikasi penjadwalan perkuliahan menggunakan *Particle Swarm Optimization* (PSO). Informasi ini akan digunakan untuk mengidentifikasi masalah. Studi pustaka dilakukan untuk memperoleh teori dan konsep yang mendasar mengenai materi yang berhubungan dalam pembuatan aplikasi penjadwalan mata kuliah menggunakan *Particle Swarm Optimization* (PSO) yaitu dengan cara mempelajari buku, artikel, jurnal dan media lainnya.

3.2 Perumusan Masalah

Dengan memanfaatkan informasi-informasi yang didapat dari penelitian pendahuluan dan studi pustaka yang telah dilakukan, maka dilakukan tahap berikutnya yaitu mengidentifikasi masalah. Pada tugas akhir ini masalah yang akan diidentifikasi adalah bagaimana membuat sebuah aplikasi yang dapat memudahkan pada penyusunan jadwal mata kuliah.

3.3 Pengumpulan Data

Pada tahap ini dilakukan pengumpulan data tentang penjadwalan mata kuliah. Semua tahap pada proses pengumpulan data tersebut diperoleh dari wawancara, observasi,dan studi pustaka.

a. Wawancara (*Interview*)

Proses wawancara dilakukan kepada Sekretaris Jurusan Teknik Informatika UIN Suska. Wawancara yang dilakukan untuk mendapatkan data mata kuliah, dosen, kelas, aturan waktu yang digunakan dan kendala yang sering dihadapi dalam penyusunan jadwal mata kuliah.

b. Observasi

Observasi dilakukan dengan melakukan pengamatan langsung terhadap alur proses penjadwalan, serta aturan-aturan yang berlaku di Jurusan Teknik Informatika UIN Suska terutama dalam proses penjadwalan mata kuliah, serta peraturan akademik yang berlaku.

c. Studi Pustaka (*Library Research*)

Studi pustaka dilakukan dengan tujuan untuk mengetahui metode apa yang akan digunakan dalam menyelesaikan permasalahan yang akan diteliti, serta mendapatkan dasar referensi yang kuat dalam menerapkan suatu metode yang akan digunakan dalam tugas akhir ini, yaitu dengan mempelajari buku, artikel, dan jurnal yang berhubungan dengan permasalahan yang akan dibahas.

3.4 Analisa Sistem

Tahap ini merupakan tahap analisa terhadap data-data yang telah dikumpulkan. Analisa sistem berguna untuk mengetahui alur proses kerja dari kerja manual agar aplikasi yang dihasilkan nantinya dapat dibuat secara maksimal. Pada tahap ini dilakukan analisa terhadap *Particle Swarm Optimization* (PSO), mulai dari pembangkitan posisi serta kecepatan partikel, evaluasi nilai *fitness* untuk mengukur tingkat kebaikan atau kesesuaian (*fitness*) suatu solusi dengan solusi yang dicari, *update velocity* (*update* kecepatan), dan *update position* (*update* posisi). Setelah dilakukan tahap analisa PSO, dilanjutkan pada tahap analisa sistem meliputi pembuatan *Data Flow Diagram* (DFD), *Entity Relationship Diagram* (ERD), dan *Flowchart*.

3.5 Perancangan Sistem

Pada tahap ini dilakukan perancangan terhadap sistem yang akan dibangun. Perancangan sistem meliputi perancangan *database*, perancangan struktur menu dan perancangan *interface.*

3.6 Implementasi

Tahap implementasi merupakan tahap penerjemahan hasil analisa ke dalam bentuk *coding* sesuai dengan hasil perancangan sistem yang dibuat. Bahasa pemrograman yang akan digunakan untuk membangun aplikasi penjadwalan mata kuliah menggunakan PSO adalah PHP dengan *database* MySQL.

3.7 Pengujian

Pengujian merupakan tahapan di mana aplikasi akan dijalankan. Tahap pengujian diperlukan untuk menjadi ukuran bahwa sistem dapat dijalankan sesuai

dengan tujuan. Metode pengujian yang digunakan yaitu *black box,* pengujian performansi, dan *User Acceptance Test.*

3.8 Kesimpulan dan Saran

Dalam tahap ini dapat ditentukan kesimpulan terhadap hasil pengujian yang telah dilakukan untuk mengetahui apakah implementasi sistem yang telah dilakukan dapat beroperasi dengan baik dan sesuai dengan tujuan yang diinginkan serta memberikan saran-saran untuk menyempurnakan dan mengembangkan penelitian selanjutnya.

BAB IV
ANALISA DAN PERANCANGAN

Analisa memegang peranan yang penting dalam membuat rincian sistem baru. Analisa merupakan langkah pemahaman permasalahan yang akan dipecahkan sebelum mengambil tindakan atau keputusan. Sedangkan perancangan adalah membuat rincian sistem hasil dari analisa menjadi suatu bentuk perancangan sistem yang mudah dimengerti oleh pengguna (*user friendly*).

4.1 Analisa Sistem Lama

Analisa sistem lama dilakukan untuk mendapatkan informasi penting dan menjadi masukan bagi sistem baru yang akan dikembangkan agar mampu mengatasi kelemahan yang terdapat pada sistem lama. Pada sistem lama, Sekretaris Jurusan Teknik Informatika UIN Suska Riau yang bertugas menyusun jadwal mengalami kesulitan dalam pembuatan jadwal perkuliahan. Hal ini disebabkan banyaknya jumlah mata kuliah, jumlah dosen, jumlah ruangan dan jumlah mahasiswa yang harus dikombinasikan agar mendapatkan jadwal yang sesuai dengan yang diinginkan.

Jadwal disusun secara manual dengan mengamati setiap ruangan, hari, dan jam yang tersedia agar tidak terjadi bentrok antar jadwal. Jadwal disusun mengikuti aturan yang berlaku pada Jurusan Teknik Informatika, yaitu:

a. Setiap mata kuliah disajikan maksimal 2 (dua) kali sehari.
b. Setiap dosen dijadwalkan mengajar 2 (dua) kali sehari.
c. Tidak terdapat perkuliahan pada jam shalat jumat.
d. Tidak terdapat perkuliahan pada jam makan siang.
e. Mata kuliah inti dijadwalkan pagi hari.
f. Dosen praktisi dijadwalkan mengajar hari Sabtu.
g. Dosen TIF dan non-TIF dijadwalkan mengajar pada hari senin-jumat.
h. Mata kuliah pilihan dijadwalkan siang hari.
i. Tidak ada perkuliahan pada hari Kamis dari pukul 08.00–12.00 bagi mahasiswa semester I, II, dan III.

j. Waktu perkuliahan yang tersedia adalah hari Senin–Rabu adalah antara pukul 08.00–16.00 WIB, dan hari Kamis–Jumat antara pukul 08.00–16.30 WIB.

4.2 Analisa Sistem Baru

Penjadwalan mata kuliah merupakan suatu masalah yang sangat kompleks. Pada Jurusan Teknik Informatikan UIN Suska menggunakan 10 (sepuluh) ruangan yang terdiri dari 8 (delapan) ruang belajar dan 2 (dua) laboratorium yang bisa digunakan untuk Kegiatan Belajar Mengajar (KBM). Misalkan, pada semester genap Tahun Ajaran 2011/2012 terdapat 121 pertemuan kuliah di mana setiap pertemuan membutuhkan waktu 2-3 jam kuliah yang harus ditentukan jadwal kuliah dan ruangannya dalam satu minggu (Senin sampai Sabtu). Setiap ruangan memiliki 54 slot dalam satu minggu perkuliahan, di mana setiap harinya terdapat 9 slot yang dapat digunakan dan setiap slot mewakili 1 jam perkuliahan (SKS). Pada analisa sistem baru, akan dibangun suatu Sistem Penjadwalan Mata Kuliah yang menerapkan metode *Particle Swarm Optimization* (PSO). Proses akan dimulai dengan membangkitkan posisi dan kecepatan dari partikel secara *random*, kemudian partikel akan berubah posisinya dari suatu perpindahan (iterasi) ke posisi lainnya berdasarkan *update* kecepatan. Dengan menggunakan nilai *fitness* akan diperoleh partikel mana yang memiliki nilai terbaik global (*global best*) saat ini dan juga dapat ditentukan posisi terbaik dari tiap partikel pada semua waktu yang sekarang dan sebelumnya, sehingga diperoleh jadwal yang optimal dari partikel-partikel tersebut.

Untuk membangun Sistem Penjadwalan Mata Kuliah perlu dilakukan analisa dan perancangan sehingga sistem yang dibangun sesuai dengan tujuan yang ingin dicapai. Analisa yang dilakukan adalah analisa data masukan (*input*), analisa data keluaran (*output*), analisa kebutuhan fungsi, dan analisa kebutuhan perangkat lunak.

4.3 Analisa Data Masukan (*Input*)

Dalam membangun suatu aplikasi penjadwalan mata kuliah menggunakan *Particle Swarm Optimization* (PSO) diperlukan data-data yang menunjang agar

sistem dapat berjalan sesuai harapan. Di mana data-data yang akan diinputkan ke sistem saling berelasi antara data yang satu dengan data yang lainnya. Data-data yang dibutuhkan sistem adalah sebagai berikut:

1. Data Mata Kuliah (MK)

   Berisi informasi mata kuliah yang disajikan untuk tiap semester, meliputi kode MK, nama MK, jumlah SKS, status MK, jenis MK, dan semester MK.

2. Data Dosen

   Berisi informasi tentang identitas dosen, meliputi kode dosen, nama dosen, dan status dosen.

3. Data Ajar

   Berisi informasi pengaturan kesediaan mengajar oleh dosen terhadap mata kuliah yang tersedia dan penentuan kelas.

4. Data Ruangan

   Berisi informasi tentang ruangan yang dipakai untuk kegiatan perkuliahan, meliputi kode ruangan, dan nama ruangan.

5. Data Hari

   Berisi informasi tentang hari yang digunakan untuk kegiatan perkuliahan, meliputi kode hari, dan nama hari.

6. Data Waktu

   Berisi informasi tentang slot waktu yang digunakan untuk kegiatan perkuliahan dalam sehari, yang meliputi kode waktu, dan slot waktu.

4.4 Analisa Data Keluaran (*Output*)

Output yang diharapkan dari aplikasi penjadwalan mata kuliah ini berupa jadwal perkuliahan bagi admin secara keseluruhan, dan jadwal bagi dosen.

4.5 Analisa Kebutuhan Fungsi

Aplikasi penjadwalan mata kuliah membutuhkan beberapa fungsi agar dapat digunakan sebagaimana mestinya oleh *user*, dan memberikan hasil yang optimal. Fungsi yang dibutuhkan oleh *user* adalah sebagai berikut:

1. Fungsi *input* data perkuliahan, terdiri dari data mata kuliah, data dosen, data ajar, data ruangan, data hari dan data jam.
2. Fungsi proses pembuatan jadwal, terdiri atas proses pembuatan jadwal, penyisipan dan pindah jadwal.
3. Fungsi pencetakan laporan, yang digunakan untuk menampilkan dan mencetak laporan jadwal perkuliahan yang dihasilkan oleh sistem.

Sedangkan fungsi-fungsi yang dibutuhkan oleh sistem adalah sebagai berikut:

1. Fungsi Pembangkitan Posisi dan Kecepatan

   Fungsi ini digunakan untuk inisialisasi dan pembuatan populasi awal dengan mengacak semua data ajar, data ruangan, data hari, dan data jam menjadi partikel-partikel.
2. Fungsi *Fitness*

   Fungsi yang digunakan untuk menyatakan seberapa baik nilai dari suatu individu ataupun solusi yang didapatkan.
3. Fungsi *Update* Kecepatan

   Fungsi ini digunakan untuk memperbarui nilai kecepatan berdasarkan nilai fungsi yang diberikan.
4. Fungsi *Update* Posisi

   Fungsi ini digunakan untuk memperbarui nilai posisi baru partikel berdasarkan nilai kecepatan perpindahan partikel.
5. Fungsi *Setting* (Pengaturan)

   Dalam fungsi ini tedapat fasilitas untuk menentukan parameter-parameter dalam PSO, seperti nilai faktor *inertia*, *learning rates (self confidence, swarm confidence*). Akan tetapi parameter ini juga memiliki nilai *default* untuk mengantisipasi bila perubahan nilai parameter menghasilkan kinerja yang kurang memuaskan.

Kondisi-kondisi *constraint* yang ditetapkan adalah sebagai berikut:

1. Ruangan kelas tidak boleh dijadwalkan lebih dari satu perkuliahan secara bersamaan.

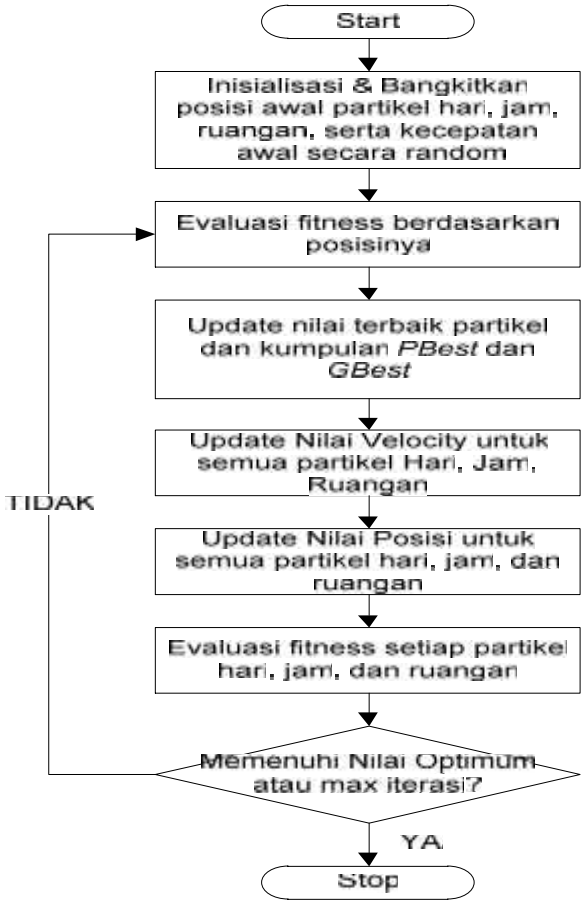
2. Perkuliahan mahasiswa tidak boleh dijadwalkan secara bersamaan.
3. Dosen tidak boleh dijadwalkan mengajar secara bersamaan.
4. Setiap mata kuliah disajikan maksimal 2 (dua) kali sehari.
5. Setiap dosen dijadwalkan mengajar 2 (dua) kali sehari.
6. Tidak terdapat perkuliahan pada jam shalat jumat.
7. Tidak terdapat perkuliahan pada jam makan siang.
8. Mata kuliah inti dijadwalkan pagi hari.
9. Dosen praktisi dijadwalkan mengajar hari Sabtu.
10. Dosen TIF dan non-TIF dijadwalkan mengajar pada hari senin-jumat.
11. Mata kuliah pilihan dijadwalkan siang hari.
12. Tidak ada perkuliahan pada hari Kamis dari pukul 08.00–12.00 bagi mahasiswa semester I, II, dan III.
13. Waktu perkuliahan yang tersedia adalah hari Senin–Rabu adalah antara pukul 08.00–16.00 WIB, dan hari Kamis–Jumat antara pukul 08.00–16.30 WIB.

4.6 Analisa Kebutuhan Perangkat Lunak

Perangkat lunak tambahan digunakan dalam pengembangan dan implementasi Aplikasi Penjadwalan Mata Kuliah Menggunakan *Particle Swarm Optimization* (PSO) adalah:

1. Macromedia Dreamweaver 8, untuk pembuatan perangkat lunak.
2. MySQL, sebagai pengolahan basis data.
3. XAMPP, sebagai *webserver*.
4. Windows XP, sebagai sistem operasi yang digunakan.

4.7 Analisa PSO dalam Penjadwalan Mata Kuliah



Gambar 4.1 Diagram alir analisa PSO dalam penjadwalan mata kuliah

Metode dalam PSO yang akan dipakai untuk penyelesaian persoalan penjadwalan mata kuliah adalah :

4.7.1 Inisialisasi dan Bangkitkan Posisi dan *Velocity* Partikel

4.7.1.1 Inisialisasi Partikel

Penginisialisasian Partikel ini adalah suatu proses bagaimana suatu jadwal kuliah dimasukkan ke dalam model Partikel yang di dalamnya mengandung beberapa basis data. Partikel yang diinisialisasi dalam penjadwalan mata kuliah ini meliputi data:

- Jam
- Ruang
- Hari
- Dosen

- Mata kuliah
- Ajar

Di bawah ini merupakan penginisialisasian partikel untuk data jam perkuliahan pada Jurusan Teknik Informatika UIN Suska Riau.

Tabel 4.1 Tabel jam

Id_Jam	Jam
1	08.00 – 08.50
2	08.50 – 09.40
3	09.40 – 10.30
4	10.30 – 11.20
5	11.20 – 12.10
6	13.00 – 13.50
7	13.50 – 14.40
8	14.40 – 15.30
9	15.30 – 16.20

Di bawah ini merupakan penginisialisasian partikel untuk data ruangan perkuliahan pada Jurusan Teknik Informatika UIN Suska Riau.

Tabel 4.2 Tabel ruangan

Id_Ruangan	Nama Ruangan
1	TIF 301
2	TIF 302
3	TIF 303
4	TIF 304
5	TIF 305
6	PSI 101
7	PSI 102
8	PSI 103
9	LAB 1
10	LAB 2

Inisialisasi Partikel selengkapnya dapat dilihat pada Lampiran B.

4.7.1.2 Pembangkitan Posisi dan Kecepatan Partikel

Dalam penjadwalan mata kuliah ini posisi dan kecepatan partikel dimisalkan sebagai slot. Posisi dan kecepatan awal ditentukan secara acak dengan batasan minimum slot dan maksimum slot. Nilai posisi dan kecepatan awal partikel diinisialisasi sama, sehingga hanya melakukan satu kali *random* untuk mendapatkan nilai kecepatan dan posisi partikel.

Pada penjadwalan ini posisi partikel diwakili oleh slot-slot, satu slot jadwal yang terdiri dari beberapa kelas merupakan satu partikel. Setiap ruangan memiliki 54 slot dalam satu minggu perkuliahan, di mana setiap harinya terdapat 9 slot yang dapat dipergunakan dan setiap slot mewakili 1 jam perkuliahan (SKS).

Pada tahap ini dilakukan proses pembangkitan posisi dan juga kecepatan partikel yang terdiri dari 4 buah partikel yang setiap partikelnya mewakili tabel ajar, tabel ruangan, tabel hari dan tabel jam (Tabel 4.6). Pembangkitan posisi dan kecepatan ini dilakukan secara acak dengan menggunakan persamaan (2.1) dan persamaan (2.2) terhadap *records* pada tabel ruangan, tabel hari dan tabel jam sebanyak jumlah *records* yang terdapat pada tabel ajar (Tabel 4.6). Nilai x_{min} (batas terendah) dan x_{max} (batas tertinggi) untuk setiap tabel adalah sebagai berikut:

Tabel 4.3 Batas bawah dan batas atas partikel hari, jam, dan ruang

Batas Bawah / Batas Atas	Tabel		
	Hari	Jam	Ruangan
x_{min}	1	1	1
x_{max}	6	9	10

Sedangkan tabel ajar sendiri sudah mengandung *fields* mata kuliah, dosen dan kelas.

4.7.1.3 Contoh Perhitungan Manual Pada Iterasi Pertama

- Posisi untuk mata kuliah A Dosen 1

  Posisi Hari

  Berdasarkan persamaan 2.1 dan persamaan 2.2 didapatkan hasil,

  $x_1^1 = 1 + rand\ (6-1) \rightarrow misal\ hasil\ random\ 0{,}1$

  = 1 + 0,1 = 1,1

  Posisi Jam

  Berdasarkan persamaan 2.1 dan persamaan 2.2 didapatkan hasil,

  $x_1^1 = 1 + rand\ (9-1) \rightarrow misal\ hasil\ random\ 7{,}1$

  $= 1 + 7{,}1 = 8{,}1$

  Posisi Ruangan

  Berdasarkan persamaan 2.1 dan persamaan 2.2 didapatkan hasil,

  $x_1^1 = 1 + rand\ (10-1) \rightarrow misal\ hasil\ random\ 2$

  $= 1 + 2 = 3$

  Jadi posisi untuk mata kuliah A dosen 1 adalah x(1.1 , 8.1 , 3.0)

  Velocity pada iterasi pertama diinisialisasi sama dengan posisi v(1.1 , 8.1 , 3.0)

Perhitungan di atas diulangi sampai sejumlah partikel, kemudian dihitung nilai *fitness* masing-masing partikel.

Berikut adalah hasil pembangkitan posisi partikel pada iterasi pertama,

Tabel 4.4 Pembangkitan posisi partikel pada iterasi pertama

Partikel 1				Partikel 2	Partikel 3	Partikel 4
Ajar				Hari	Jam	Ruangan
Id	Makul	Dosen	Kelas			
1	1	26	A	1,1	8,1	3
2	1	26	B	2,6	4,6	4
3	1	26	C	2,1	4,6	4,5
4	1	25	D	2,8	6,4	5,2

Perhitungan manual pembangkitan posisi secara lengkap dapat dilihat pada Lampiran C.

Berikut adalah ilustrasi pembangkitan partikel dalam bentuk jadwal,

Tabel 4.5 Ilustrasi pembangkitan partikel dalam bentuk jadwal

Hari	Jam	Ruangan									
		1	2	3	4	5	6	7	8	9	10
1 1	1										
	2										
	3										
	4										
	5										
	6										
	7										
	8			Makul A Dosen 1							
	9										
2 2	1										
	2										
	3										
	4				Makul B Dosen 2 Makul C Dosen 3						
	5										
	6					Makul D Dosen 4					
	7										
	8										
	9										

4.7.2 Evaluasi Fungsi *Fitness*

Proses ini dilakukan dengan memperhatikan *constraint* yang telah ditetapkan sebelumnya. Setiap partikel akan diperiksa satu persatu dan dibandingkan dengan partikel lainnya sesuai *constraint*. Dalam penjadwalan ini nilai *fitness* menentukan banyaknya pelanggaran *constraint* yang harus dioptimasi. Ketika terjadi pelanggaran terhadap *constraint* maka nilai pelanggaran akan ditotalkan untuk satu iterasi tersebut.

Untuk mencari nilai *fitness* dalam kasus penjadwalan mata kuliah ini, di mana kita ingin meminimalkan jumlah pelanggaran terhadap *constraint* maka dapat digunakan persamaan 2.5.

Semakin kecil jumlah pelanggaran yang terjadi, maka nilai *fitness* yang dihasilkan semakin besar. Sebaliknya semakin besar jumlah pelanggaran yang terjadi, maka nilai *fitness* yang dihasilkan semakin kecil. Partikel dengan nilai *fitness* terbesar merupakan partikel yang terbaik.

Misalkan, jumlah pelanggaran yang terjadi pada partikel setelah pembangkitan posisi dan kecepatan partikel adalah 10. Maka nilai *fitness*-nya dapat dihitung dengan persamaan 2.5:

$f = 1/(10+0{,}001) = 0{,}099 \approx 0{,}1$

4.7.3 Menentukan *Local Best* dan *Global Best*

a. *Local Best* (p^i)

Local Best (p^i) pada iterasi pertama diset sama dengan posisi awal setelah pembangkitan. Misalnya pada mata kuliah A, dosen 1, dengan posisi harinya adalah 1.1, maka nilai *local best* (p^i) diset 1.1. Demikian seterusnya hingga setiap partikel diset sama dengan posisi awal partikel. Sedangkan untuk iterasi selanjutnya, maka akan dibandingkan antara posisi sekarang (x_k^i) dengan *local best* (p^i). Jika posisi sekarang (x_k^i) lebih kecil atau sama dengan *local best* (p^i), maka *local best* (p^i) akan di-*update* dengan nilai posisi sekarang (x_k^i).

b. *Global Best* (p_k^g)

Setelah didapat nilai *fitness* terbaik pada iterasi pertama, maka langkah selanjutnya adalah menentukan partikel terbaik dalam satu iterasi, yaitu partikel dengan nilai *fitness* paling kecil dari partikel-partikel lain dalam satu iterasi. Setelah didapat nilai *fitness* pada iterasi pertama adalah 0.1, maka nilai *global best* (p_k^g) diset 0.1 untuk setiap partikel. Partikel terbaik tersebut kemudian disimpan sebagai *global best particle*. Sedangkan untuk iterasi selanjutnya, maka akan dibandingkan antara posisi sekarang (x_k^i) dengan *global best* (p_k^g). Jika posisi sekarang (x_k^i)

lebih kecil atau sama dengan *global best* (p_k^g), maka *global best* (p_k^g) akan di-*update* dengan nilai posisi *global* sekarang (x_k^g).

4.7.4 Proses *Update Velocity* dan Posisi

Untuk mencari *update velocity* maka kita menggunakan persamaan 2.3 dan untuk mencari *update* posisi kita menggunakan persamaan 2.4. Diberikan parameter ujicoba sebagai berikut:

- C1 = 1.5.
- C2 = 1.5.
- *w* = 0.5.

Parameter uji coba ini diambil dari hasil penelitian yang dilakukan oleh Dian Ariani (Ariani, 2011) yang mana dengan parameter ini dapat menghasilkan rata-rata jadwal yang lebih optimal.

- Posisi untuk mata kuliah A Dosen 1

Posisi Hari

Misal hasil rand 1 = 0.3 dan rand 2 = 0.2

v_2^1 = 0.5 * 1.1 + 1.5 * 0.3 (1.1 – 1.1) + 1.5 * 0.2 (1.1 – 1.1)

= 0.55

x_2^1 = 1.1 + 0.55

= 1.65

Posisi Jam

Misal hasil rand 1 = 0.3 dan rand 2 = 0.2

v_2^1 = 0.5 * 8.1 + 1.5 * 0.3 (8.1 – 8.1) + 1.5 * 0.2 (8.1 – 8.1)

= 4.05

x_2^1 = 8.1 + 4.05

= 12,2

Posisi Ruangan

Misal hasil rand 1 = 0.3 dan rand 2 = 0.2

v_2^1 = 0.5 * 3 + 1.5 * 0.3 (3 – 3) + 1.5 * 0.2 (3 – 3)

= 1.5

$x_2^1 = 3 + 1.5$

$= 4.5$

Berikut ini ditampilkan hasil perhitungan manual *update velocity* dan posisi pada iterasi pertama.

Tabel 4.6 Hasil perhitungan *update velocity* dan posisi pada iterasi pertama

Partikel				**1**	**2**	**Partikel 2**	**Partikel 3**	**Partikel 4**
Ajar								
d	Makul	Dosen	elas			Hari	Jam	Ruangan
1	1	26	A	0.3	0.2	1.7	12.2	4.5
2	1	26	B	0.3	0.4	3.9	6.9	6.0
3	1	26	C	0.1	0.2	3.2	4.9	7.8
4	1	25	D	0.1	0.3	4.2	9.6	7.8

Perhitungan manual *Update Velocity* dan Posisi secara lengkap dapat dilihat pada Lampiran C.

Ilustrasi *update velocity* dan posisi dalam bentuk jadwal, dapat dilihat pada Tabel 4.7.

Langkah selanjutnya adalah hitung kembali nilai *fitness*-nya. Jika posisi semua partikel menuju ke satu nilai yang sama, maka ini disebut konvergen. Jika belum konvergen maka langkah 2 diulang dengan memperbaharui iterasi $i = i + 1$, dengan cara menghitung nilai baru dari p^i dan p_k^g. Proses iterasi ini dilanjutkan sampai semua partikel menuju ke satu titik solusi yang sama. Biasanya akan ditentukan dengan kriteria penghentian (*stopping condition*), misalnya jumlah selisih solusi sekarang dengan solusi sebelumnya sudah sangat kecil.

Tabel 4.7 Ilustrasi *update velocity* dan posisi dalam bentuk jadwal

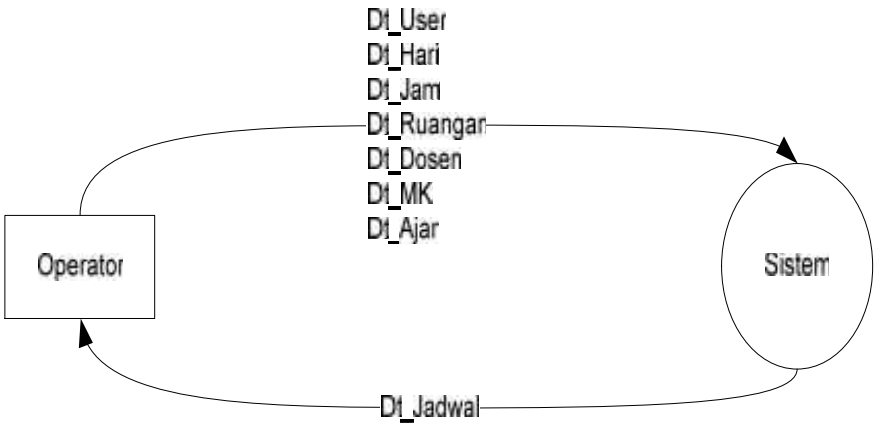
Hari	Jam	Ruangan									
		1	2	3	4	5	6	7	8	9	10
1 1	1										
	2										
	3				Makul A Dosen 1						
	4										
	5										
	6										
	7										
	8										
	9										
2 2	1										
	2										
	3										
	4										
	5										
	6										
	7										
	8										
	9										
3 3	1										
	2										
	3										
	4							Makul C Dosen 3			
	5										
	6						Makul B Dosen 2				
	7										
	8										
	9										
4 4	1										
	2										
	3										
	4										
	5										
	6										
	7										
	8										
	9							Makul D Dosen 4			

4.8 Perancangan Sistem

Perancangan sistem dalam membangun sistem penjadwalan mata kuliah pada Jurusan Teknik Informatika terdiri dari, *Context Diagram* dan DFD, perancangan *database* (struktur basis data), serta perancangan antarmuka (*user interface*).

4.8.1 *Context Diagram* (Diagram Konteks)

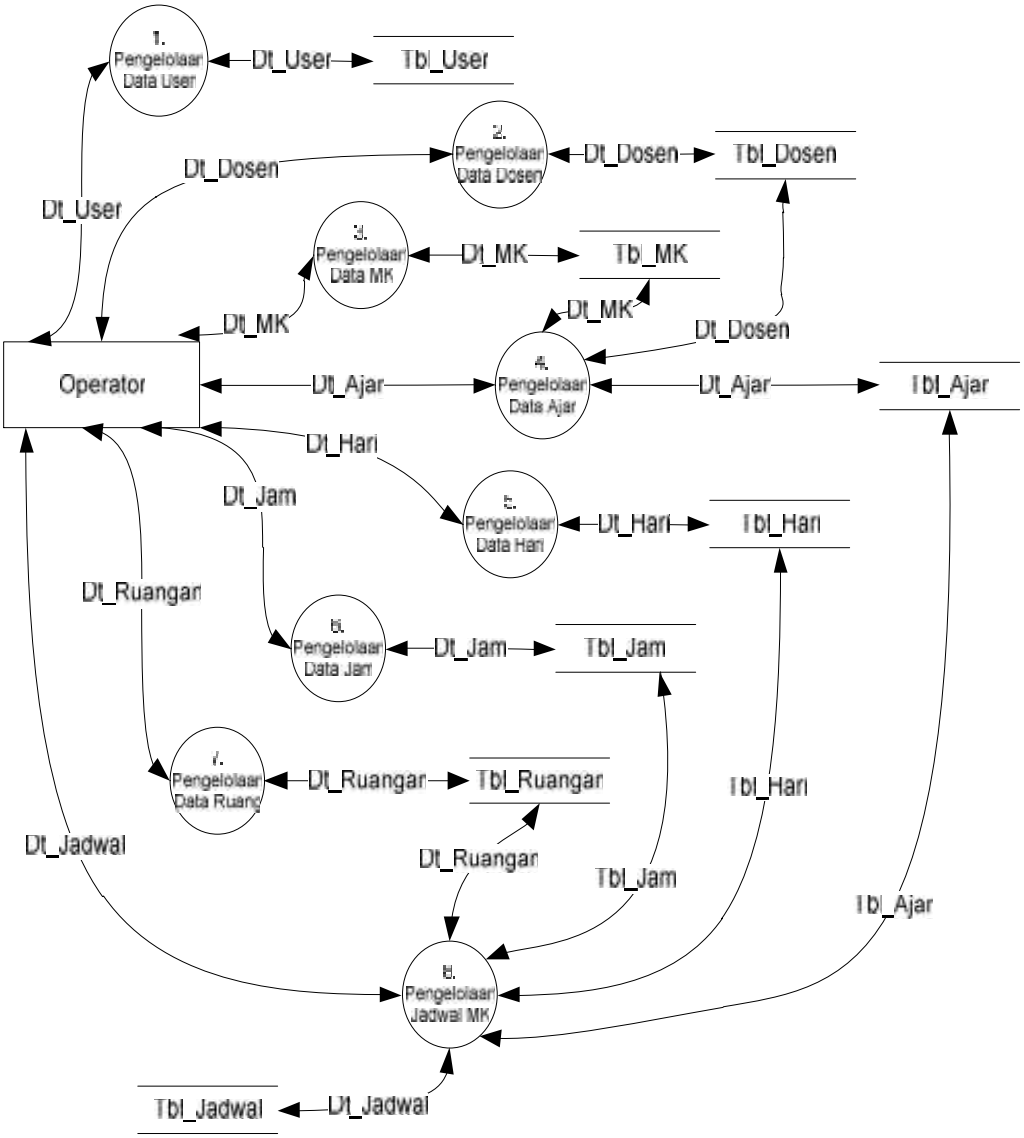
Diagram konteks merupakan level dasar DFD (level 0) yang digunakan untuk menggambarkan proses kerja suatu sistem secara umum. Berikut ini merupakan gambar diagram konteks yang akan dibangun seperti Gambar 4.2 di bawah ini.



Gambar 4.2 Diagram konteks

Pada diagram konteks di atas, entitas luar yang berinteraksi dengan sistem adalah Operator. Entitas (terminator) yang dimaksud pada DFD adalah yang memberikan sumber data ke sistem atau menerima info data dari sistem. Entitas mewakili lingkungan luar dari sistem, tetapi mempunyai pengaruh terhadap sistem yang sedang dikembangkan. Sehingga, pengguna sistem (*user*) dapat mengetahui dengan lingkungan mana saja sistem ini berhubungan. Yang menjadi Operator dalam sistem ini adalah Sekretaris Jurusan (Sekjur) Teknik Informatika UIN Suska.

Berikut ini merupakan Gambar 4.3 DFD level 1 dari sistem.



Gambar 4.3 DFD level 1

Dari Gambar 4.3 dapat dijelaskan proses DFD level 1 dan aliran datanya pada tabel 4.8 dan 4.9 di bawah ini.

Tabel 4.8 Proses DFD level 1

No. Proses	Nama	Deskripsi
1	Pengelolaan Data User	Proses yang melakukan pengelolaan data user ang merupakan pengguna sistem
2	Pengelolaan Data Dosen	Proses yang melakukan pengelolaan data Dosen yang dimasukkan oleh operator sebagai informasi dalam penjadwalan
3	Pengelolaan Data MK	Proses yang melakukan pengelolaan data mata kuliah yang dimasukkan oleh operator sebagai informasi dalam penjadwalan
4	Pengelolaan Data Ajar	Proses yang melakukan pengelolaan data ajar yang dimasukkan oleh operator sebagai informasi dalam penjadwalan
5	Pengelolaan Data Hari	Proses yang melakukan pengelolaan data hari yang merupakan informasi jumlah hari yang digunakan untuk kegiatan perkuliahan selama seminggu
6	Pengelolaan Data Jam	Proses yang melakukan pengelolaan data jam yang digunakan untuk kegiatan perkuliahan perhari
7	Pengelolaan Data Ruang	Proses yang melakukan pengelolaan data ruangan yang dimasukkan oleh operator sebagai informasi tempat dilangsungkannya perkuliahan
8	Pengelolaan Jadwal	Proses yang melakukan pembuatan jadwal, penyisipan dan pindah jadwal

Di bawah ini merupakan tabel aliran data DFD Level 1 Aplikasi Penjadwalan Mata Kuliah.

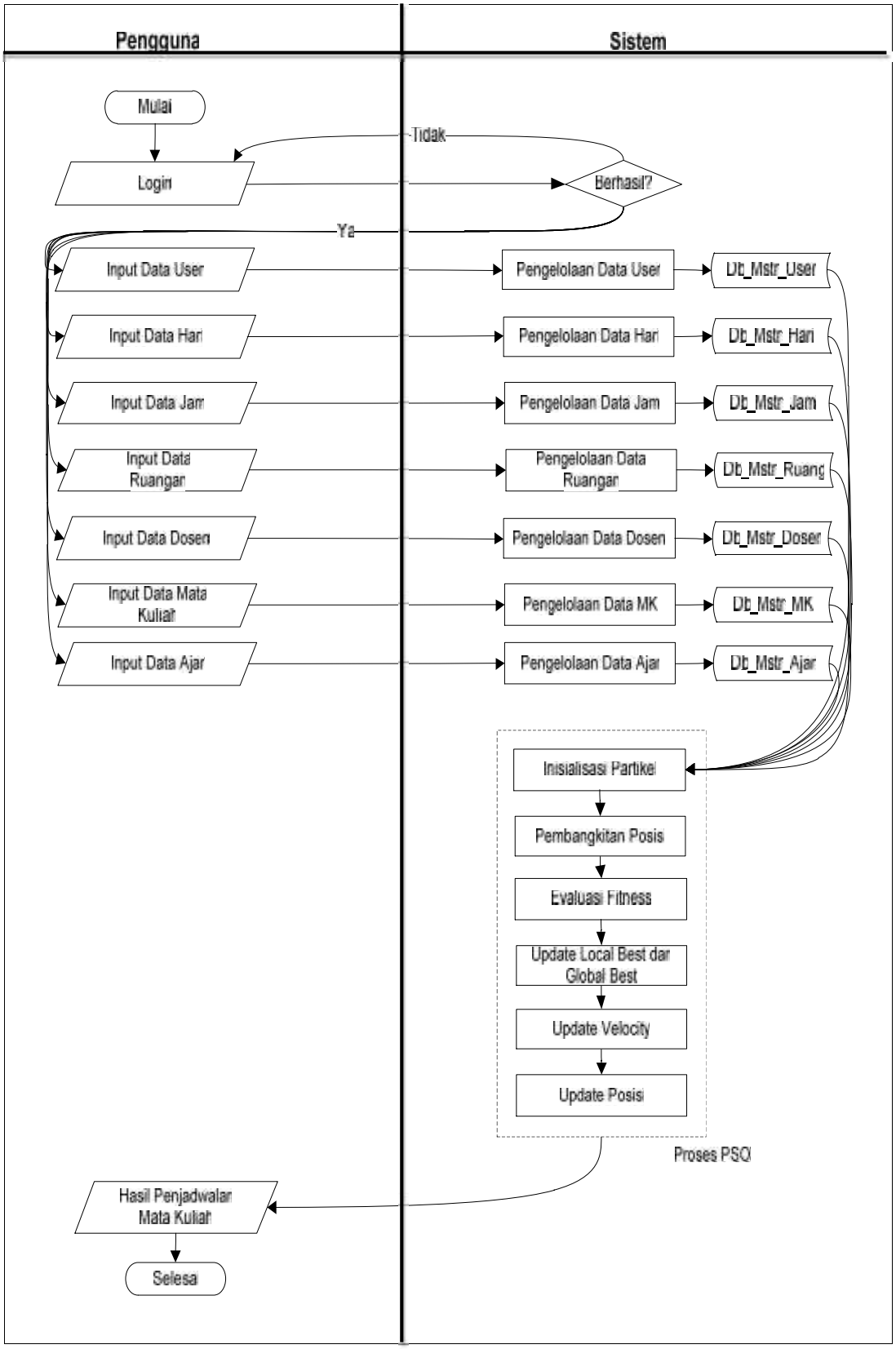
Tabel 4.9 Aliran data DFD level 1

Nama	Deskripsi
Dt_User	Data yang meliputi pengolahan data user dalam basis data
Dt_Dosen	Data yang meliputi data dosen yang mengajar semua bidang mata kuliah pada Jurusan Teknik Informatika
Dt_MK	Data yang meliputi data mata kuliah semua angkatan pada semester ganjil dan genap
Dt_Ajar	Data yang meliputi data mata kuliah, data dosen dan kelas
Dt_Hari	Data yang meliputi hari selama seminggu yang digunakan untuk kegiatan perkuliahan
Dt_Jam	Data yang meliputi banyaknya jam yang digunakan untuk kegiatan perkuliahan dalam sehari
Dt_Ruangan	Data yang meliputi bdata ruangan yang digunakan untuk perkuliahan
Dt_Jadwal	Merupakan laporan jadwal perkuliahan hasil pengolahan data MK, data dosen, data ruangan, data kelas dan data hari menggunakan Algoritma PSO

Untuk Data Flow Diagram (DFD) yang lebih rinci dapat dilihat pada Lampiran D.

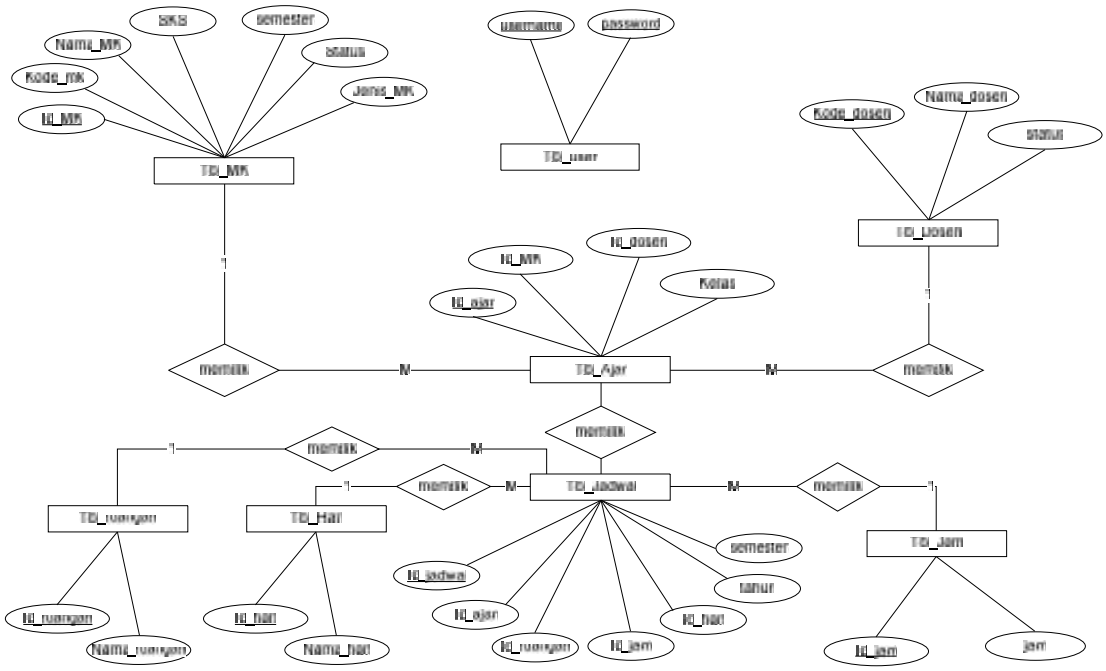
4.8.2 *Flowchart* Sistem

Flowchart sistem mendeskripsikan proses aliran sistem yang terjadi dimulai dari awal menggunakan sistem hingga selesai. Pada Gambar 4.4 dapat digambarkan *flowchart* sistem yang dibangun.



Gambar 4.4 *Flowchart* sistem

4.8.3 Entity Relationship Diagram (ERD)



Gambar 4.5 *Entity Relationship Diagram* (ERD)

4.9 Perancangan Basis Data

Perancangan basis data merupakan transformasi model data yang dihasilkan oleh proses analisa menjadi struktur data yang dibutuhkan perangkat lunak pada saat implementasi. Perancangan pada sistem ini dibutuhkan basis data yang digunakan untuk menyimpan data-data. Basis data yang digunakan bertipe basis data relasional yang terdiri dari beberapa tabel. Hasil dari perancangan basis data berupa struktur basis data. Masing-masing dari hasil perancangan basis data tersebut dapat dideskripsikan sebagai berikut:

4.9.1 Struktur Basis Data

Perancangan struktur basis data menggambarkan deklarasi dari *fields* data yang digunakan dalam perancangan aplikasi penjadwalan mata kuliah. Berikut ini merupakan perancangan struktur basis data dari masing-masing tabel.

Tabel *user* digunakan untuk menampung data pengguna aplikasi ini. Tabel berikut merupakan struktur tabel *user*.

1. Tabel User

   Nama : *User*

   Deskripsi : Berisi data-data yang digunakan untuk menampung data pengguna yang akan menggunakan sistem ini

   Primary Key : *Username* dan *Password*

Tabel 4.10 Struktur tabel *user*

Nama Field	**Type dan Length**	**Deskripsi**	**Null**
*username**	Text (30)	*username* pengguna (*Primary Key*)	Not Null
*password**	Text (32)	*password* pengguna (*Primary Key*)	Not Null

2. Tabel Mata_kuliah

   Nama : Mata Kuliah

   Deskripsi : Berisi data-data mata kuliah yang diajarkan di Jurusan Teknik Informatika

   Primary Key : id_mk

Tabel 4.11 Struktur tabel mata kuliah

Nama Field	**Type dan Length**	**Deskripsi**	**Null**
id_mk*	Varchar (20)	Id mata kuliah (*Primary Key*)	Not Null
nama_mk	Varchar (20)	Nama mata kuliah	Not Null
SKS	Integer (1)	Jumlah SKS	Not Null
status	Teks (7)	Mata kuliah wajib atau pilihan	Not Null
semester	Teks (7)	Nilai semester tiap-tiap mata kuliah	Not Null
jenis	Teks (9)	Jenis mata kuliah kelas atau praktikum	Not Null

3. Tabel Dosen

   Nama : Dosen

   Deskripsi : Berisi data-data dosen yang mengajar di Jurusan Teknik Informatika

   Primary Key : id_dosen

Tabel 4.12 Struktur tabel dosen

Nama Field	**Type dan Length**	**Deskripsi**	**Null**
kode_dosen*	varchar (5)	Singkatan / inisial nama dosen	Not null
nama_dosen	Varchar (70)	Nama Dosen	Not null
status	Varchar (20)	Status Dosen : Dosen TIF, Dosen Praktisi, dan Dosen non-TIF	Not null

4. Tabel Ajar

   Nama : Ajar

   Deskripsi : Berisi data-data mata kuliah yang diajarkan oleh dosen dan kelasnya

   Primary Key : id_ajar

Tabel 4.13 Struktur tabel ajar

Nama Field	**Type dan Length**	**Deskripsi**	**Null**
id_ajar*	Integer (4)	Id Ajar (Primary Key)	Not null
id_mk	Integer (3)	Id mata kuliah yang akan diajarkan	Not null
id_dosen	Integer (3)	Id dosen yang mengajar	Not null
kelas	Text (1)	Kelas yang akan diajar	Not null

5. Tabel Ruangan

   Nama : Ruangan

   Deskripsi : Berisi data ruangan yang akan digunakan untuk kegiatan perkuliahan

   Primary Key : Id_ruangan

Tabel 4.14 Struktur tabel ruangan

Nama Field	**Type & Length**	**Deskripsi**	**Null**
Id_ruangan*	Integer (2)	Id ruangan (Primary Key)	Not null
Nama_ruangan	Text (8)	Nama ruangan	Not null

6. Tabel Hari

   Nama : Hari

   Deskripsi : Berisi banyaknya hari yang digunakan untuk kegiatan perkuliahan

   Primary Key : Id_hari

Tabel 4.15 Struktur tabel hari

Nama Field	**Type dan Length**	**Deskripsi**	**Null**
Id_hari*	Integer (2)	Id hari (Primary Key)	Not null
Nama_hari	Text (8)	Nama hari	Not null

7. Tabel Jam

   Nama : Jam

   Deskripsi : Berisi pembagian jam yang digunakan untuk kegiatan kuliah

   Primary Key : Id_jam

Tabel 4.16 Struktur tabel jam

Nama Field	Type dan Length	Deskripsi	Null
Id_jam*	Integer (2)	Id jam (Primary Key)	Not null
Jam	Time	Nama Jam	Not null

8. Tabel Jadwal

   Nama : Jadwal

   Deskripsi : Berisi jadwal perkuliahan yang telah dihasilkan oleh Algoritma PSO

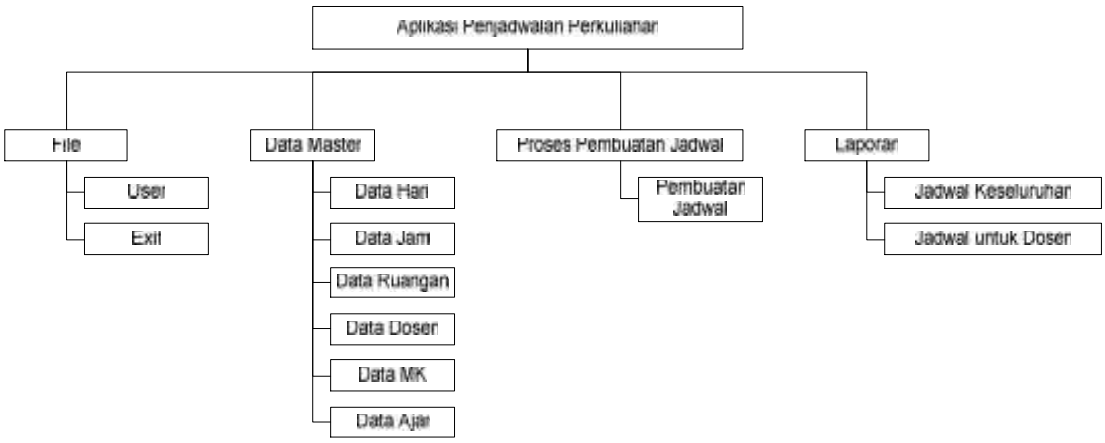
   Primary Key : Id_jadwal

Tabel 4.17 Struktur tabel jadwal

Nama Field	Type dan Length	Deskripsi	Null
Id_jadwal*	Integer (3)	Id jadwal	Not null
Id_ajar	Integer (4)	Id ajar	Not null
Id_ruangan	Integer (2)	Id ruangan	Not null
Id_hari	Integer (2)	Id hari	Not null
Id_jam	Integer (2)	Id jam	Not null
Tahun	Integer (4)	Tahun ajaran	Not null
Semester	Text (1)	Jadwal semester ganjil atau genap	Not null

4.10 Perancangan Struktur Menu

Tujuan perancangan adalah untuk membuat panduan pada tahap implementasi mengenai rancangan dari sistem yang akan dibuat, supaya implementasi dapat dilakukan secara modular tetapi tetap konsisten. Masalah yang akan diselesaikan adalah penjadwalan mata kuliah. Perancangan antar muka secara diagram dapat dilihat pada Gambar 4.6



Gambar 4.6 Struktur menu

4.11 Perancangan Tampilan Sistem

Agar aplikasi ramah pengguna, maka perlu dirancang tampilan-tampilan yang mudah dimengerti oleh pengguna, sehingga pengguna mudah menggunakan aplikasi ini. Berikut ini beberapa rancangan tampilan yang sesuai dengan perancangan struktur menu yang dibuat. Untuk spesifikasi perancangan tampilan sistem / antarmuka yang lebih rinci dapat dilihat pada Lampiran E.

Gambar 4.7 Desain utama aplikasi

Tabel 4.18 Spesifikasi objek tampilan utama

Nama Objek	Jenis	Keterangan
Menu File	MenuBar	Form untuk pengolahan data pengguna
Menu Data Master	MenuBar	Form pengelolaan data master (tambah, ubah, hapus)
Menu Proses Pembuatan Jadwal	MenuBar	Form untuk proses pembuatan jadwal, penyisipan dan pindah jadwal
Menu Laporan	MenuBar	Form untuk mencetak laporan

4.12 Perancangan *Output* Jadwal

Jadwal yang dihasilkan oleh aplikasi jadwal perkuliahan diharapkan dapat dicetak. Rancangan antarmuka jadwal perkuliahan adalah sebagai berikut:

JADWAL MATA KULIAH TEKNIK INFORMATIKA

SEMESTER xxxx TAHUN xxxx

Hari	Jam	Ruangan										
		0	1	2	3	4	5	6	7	8	9	10
xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx
xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx

Berikut ini merupakan rancangan *output* jadwal untuk Dosen

JADWAL MENGAJAR

Hari	Jam	Mata Kuliah	Ruangan	Kelas	SKS	Semester
Xx	xx	xx	xx	xx	xx	xx

BAB V
IMPLEMENTASI DAN PENGUJIAN

5.1 Implementasi

Implementasi merupakan tahap dilakukan pengkodean hasil dari analisa dan perancangan ke dalam sistem, sehingga akan diketahui apakah sistem yang dibuat telah menghasilkan tujuan yang diinginkan.

5.1.1 Lingkungan Implementasi

Lingkungan implementasi adalah lingkungan di mana aplikasi ini dikembangkan. Lingkungan implementasi sistem ada dua yaitu lingkungan perangkat keras dan lingkungan perangkat lunak, dengan spesifikasi sebagai berikut:

1. Perangkat Keras

   Perangkat keras yang digunakan mempunyai spesifikasi sebagai berikut:

   a. *Processor* : Intel Dual Core
   b. *Memory* : 2 GB
   c. *Hardisk* : 320 GB

2. Perangkat Lunak

   Perangkat lunak yang digunakan adalah sebagai berikut:

   1. *Operating System* : *Windows XP Professional SP 3*
   2. Bahasa Pemrograman : *PHP*
   3. *Database* : *MySQL*
   4. *Webserver* : Apache

5.1.2 Implementasi Model Persoalan

Model persoalan pada sistem ini akan menghasilkan rekomendasi penjadwalan mata kuliah optimal berdasarkan perhitungan nilai posisi oleh perhitungan PSO terhadap *constraint* yang ditetapkan. Penggunaan sistem sesuai model persoalan yang telah dijelaskan pada BAB IV sebelumnya. Adapun tampilan menu sistem ini sebagai berikut:

5.1.2.1 Tampilan Menu *Login*

Menu *login* pada sistem ini berguna untuk validasi data pengguna. Sebelum masuk ke menu utama, pengguna harus meng*input*kan nama pengguna dan kata sandinya. Setelah mengklik tombol masuk, sistem mengecek *database* dengan data *login* yang di*input*kan oleh pengguna. Jika data yang di*input*kan benar, akan masuk ke tampilan menu utama Tampilan menu *login* dapat dilihat pada Gambar 5.1 di bawah ini.

Gambar 5.1 Tampilan menu *login valid*

5.1.2.2 Tampilan Menu Utama

Tampilan menu utama dapat diakses jika menu *login* dinyatakan *valid*.

Gambar 5.2 Tampilan menu utama

Menu yang terdapat pada menu utama, yaitu:

1. Menu Home

   Digunakan untuk kembali ke menu utama
2. Menu Data Master

   Digunakan untuk menambah, mengubah dan menghapus data hari, data jam, data ruangan, data dosen, data mata kuliah, dan data ajar.

3. Menu Proses Penjadwalan.

   Digunakan untuk memulai proses pembuatan jadwal mata kuliah, dan pemindahan jadwal.

4. Menu Laporan

   Digunakan untuk membuat laporan jadwal mata kuliah dan jadwal mengajar bagi dosen.

Implementasi secara rinci dapat dilihat pada lampiran E.

5.2 Pengujian Sistem

Pengujian sistem dilakukan terhadap program yang telah dirancang. Pengujian sistem dilakukan dengan tujuan untuk menjamin sistem yang dibangun sesuai dengan hasil analisa dan perancangan sehingga dapat dibuat satu kesimpulan akhir.

5.2.1 Lingkungan Pengujian Sistem

Pengujian sistem ini dilakukan pada lingkungan perangkat lunak dan perangkat keras yang sama dengan lingkungan implementasi.

5.2.2 Rencana Pengujian

Rencana pengujian sistem terbagi dalam dua kategori yaitu pengujian sistem dan pengujian performansi *Particle Swarm Optimization* (PSO) dalam menghasilkan jadwal perkuliahan. Pengujian ini dilakukan dengan melibatkan pihak lain sebagai pengguna sistem, yaitu:

1. Nama : Yudi Nasrendra
   NIM : 10655005226
   Jenis Kelamin : Laki-Laki
   Status : Mahasiswa Teknik Elektronika UIN Suska Riau
2. Nama : Alfi Syahri
   Jenis Kelamin : Laki-Laki
   Status : Alumni Teknik Elektronika UIN Suska Riau

5.3 Deskripsi dan Hasil Pengujian

Model atau cara pengujian pada sistem ini ada tiga cara yaitu:

1. Menggunakan *Black Box* (Keterangan selanjutnya pada 5.3.1)
2. Menggunakan Performansi (Keterangan selanjutnya pada 5.3.2)
3. Menggunakan *User Acceptence Test* (Keterangan selanjutnya pada 5.3.3)

5.3.1 Pengujian Sistem dengan *Black Box*

Pengujian *Black Box* adalah pengujian yang fokus pada persyaratan fungsional perangkat lunak. Pengujian ini juga dilakukan untuk menguji kebenaran output yang dihasilkan oleh aplikasi. Hasil pengujian sistem yang dilakukan dengan menggunakan *black box* adalah:

Tabel 5.1 Hasil pengujian *Black Box*

No. Uji	Butir Uji	Hasil Pengujian
Uji 1	Pengujian Modul Login	Diterima
Uji 2	Pengujian Modul Menu *Input* Data Hari	Diterima
Uji 3	Pengujian Modul Menu *Input* Data Jam	Diterima
Uji 4	Pengujian Modul Menu *Input* Data Ruangan	Diterima
Uji 5	Pengujian Modul Menu *Input* Data Dosen	Diterima
Uji 6	Pengujian Modul Menu *Input* Data Mata Kuliah	Diterima
Uji 7	Pengujian Modul Menu *Input* Data Ajar	Diterima
Uji 8	Pengujian Modul Menu Proses	Diterima

Pengujian *black box* selengkapnya dapat dilihat pada lampiran G.

5.3.2 Pengujian Performansi

Pengujin performansi yang dilakukan dengan pengujian menggunakan parameter nilai *default* dan parameter inputan *user*.

Tabel 5.2 Parameter perhitungan dengan nilai *default*

C1	*C2*	*w*	Iterasi Maksimum
1,5	1,5	0,5	1000

Tabel 5.3 Pengujian performansi dengan parameter nilai *default*

Pengujian ke-	Jumlah Iterasi	Hasil Pengujian
1	40	Berhasil
2	28	Berhasil
3	32	Berhasil
4	39	Berhasil
5	29	Berhasil
6	37	Berhasil
7	23	Berhasil
8	29	Berhasil
9	45	Berhasil
10	23	Berhasil

Tabel 5.4 Pengujian performansi dengan parameter inputan *user*

Pengujian ke-	C1	C2	*w*	Iterasi	Hasil Pengujian
1	1	2	0,6	38	Berhasil
2	3	2	0.6	48	Berhasil
3	0.7	1.4	0.4	40	Berhasil
4	3	1	0.5	28	Berhasil
5	2	0.5	0.9	34	Berhasil
6	2	1	0.8	41	Berhasil
7	4	1	0.5	32	Berhasil
8	1	4	0.5	37	Berhasil
9	5	1	0.4	31	Berhasil
10	1	7	0.7	42	Berhasil

5.3.3 Pengujian Sistem dengan *User Acceptence Test*

User Acceptance Test merupakan pengujian yang dilakukan dengan meminta persetujuan dari *user* terhadap *output* yang dihasilkan oleh aplikasi penjadwalan mata kuliah ini. Responden yang melakukan pengujian yaitu Ketua Jurusan atau Sekretaris Jurusan Teknik Informatika UIN Suska Riau. Pengujian *User Acceptance Test* selengkapnya dapat dilihat pada lampiran H.

5.4 Hasil Pengujian

Hasil yang diperoleh dari pengujian sistem ini adalah sebagai berikut:

1. Berdasarkan pengujian menggunakan *Black box*, seluruh menu dan *button* pada sistem penjadwalan perkuliahan ini berfungsi dengan baik.
2. Berdasarkan pengujian performansi, baik yang menggunakan parameter nilai *default* maupun yang menggunakan nilai inputan dari *user*, sistem ini berhasil menghasilkan solusi penjadwalan mata kuliah yang sudah tidak terdapat bentrok terhadap hari, jam, dan ruangan perkuliahan.
3. Walaupun berhasil menghasilkan jadwal perkuliahan yang sudah tidak terdapat bentrokan, namun tidak berhasil memenuhi dari segi kualitas yaitu jam dimulainya perkuliahan yang difokuskan pada jam-jam yang efektif.
4. Dari hasil pengujian, pada sistem ini masih ditemukan mata kuliah teori yang menempati ruangan laboratorium.

5.5 Kesimpulan Pengujian

Berdasarkan pengujian yang telah dilakukan dapat diambil kesimpulan. Adapun kesimpulan dari pengujian di atas sebagai berikut.

1. Seluruh menu dan *button* pada sistem penjadwalan perkuliahan berfungsi dengan baik.
2. Penggunaan sistem ini menghasilkan jadwal dengan waktu yang relatif lebih cepat bila dibandingkan dengan penjadwalan secara manual dan hasilnya cukup memuaskan.
3. Walaupun berhasil menghasilkan jadwal perkuliahan yang sudah tidak terdapat bentrokan, namun tidak berhasil memenuhi dari segi kualitas yaitu jam dimulainya perkuliahan yang difokuskan pada jam-jam yang efektif.

BAB VI
PENUTUP

6.1 Kesimpulan

Kesimpulan dari penelitian Tugas Akhir ini adalah sebagai berikut:

1. Aplikasi penjadwalan mata kuliah menggunakan *Particle Swarm Optimization* (PSO) berhasil dirancang dan dibangun untuk menghasilkan jadwal perkuliahan di Jurusan Teknik Informatika UIN Suska Riau walaupun tidak semua *constraint* yang ditetapkan oleh pihak Jurusan terpenuhi.
2. Walaupun berhasil menghasilkan jadwal perkuliahan yang sudah tidak terdapat bentrokan, namun tidak berhasil memenuhi dari segi kualitas yaitu jam dimulainya perkuliahan yang difokuskan pada jam-jam yang efektif.
3. Aplikasi penjadwalan mata kuliah ini mampu menangani proses input data, melakukan proses pembuatan jadwal perkuliahan secara otomatis dan menghasilkan jadwal yang dapat ditampilkan dan dapat pula dicetak.
4. Aplikasi penjadwalan mata kuliah menggunakan *Particle Swarm Optimization* (PSO) memiliki kekurangan yaitu *constraint* yang tidak bisa berubah sesuai kondisi.

6.2 Saran

Dari hasil pembahasan yang telah dilakukan, dapat disarankan untuk penelitian selanjutnya agar:

1. Penelitian mengenai performansi algoritma masih sangat dibutuhkan lebih lanjut pada bidang aplikasi lainnya, sehingga mampu memberikan kontribusi pada perkembangan algoritma tersebut.
2. Untuk pengembangan selanjutnya, *constraint* dibuat dapat diubah sesuai kebutuhan dan persyaratan pembuatan jadwal yang berlaku.
3. Aplikasi menghasilkan jadwal yang memenuhi kebutuhan akan jadwal yang berkualitas, yaitu slot waktu yang ideal untuk proses penjadwalan dapat dipergunakan dengan efektif.

DAFTAR PUSTAKA

Ariani, Dian, dkk. "*Optimasi Penjadwalan Mata Kuliah di Jurusan Teknik Informatika PENS Dengan Menggunakan Algoritma Particle Swarm Optimization.*" [Online] available *at* http://www.eepis-its.edu/uploadta/downloadmk.php?id=1235, tanggal akses 31 Oktober 2011

Bambrick, Leon. "*Lecture Timetabling Using Genetic Algorithms."* The University of Queensland, Queensland. 1997.

Betrianis, dan Putu Teguh Aryawan. "Penerapan Algoritma Tabu Search Dalam Penjadwalan Job Shop." *Makara Teknologi*. Vol. 7, No. 3, halaman. 107 - 112, 2003.

Burke, Edmund K., et al. "A Graph - Based Hyper - Heuristic for Educational Timetabling Problems." *European Journal of Operational Research*, halaman 1 - 34, 2006.

Burke, Edmund K., et al. "Case - Based Heuristic Selection for Timetabling Problems." *Journal of Scheduling*, 9:99 - 113, halaman 1 - 33, 2006.

Chauduri, Arindam, dan Kajal De. "Fuzzy Genetic Heuristic For University Course Timetable Problem*."* *Int. J. Advance. Soft Comput. Appl*, Vol. 2, No. 1, halaman. 100 - 123, 2010.

Eberhart, R. C., dan Y. Shi. *"Particle Swarm Optimization : Development, Applications, and Resources*." [Online] available at http://www.nici.kun.nl/~aiweb/aicourses/mki44/slides/SwarmIntelligence/literature/Eberhart01%20 - %20PSO.pdf, tanggal akses 31 Oktober 2011

Jusuf, Heni. "Pewarnaan Graph pada Simpul Untuk Mendeteksi Konflik Penjadwalan Kuliah." *SNATI*, ISSN 1907-5022, 2009.

Kusumadewi, Sri, dan Hari Purnomo. "*Penyelesaian Masalah Optimasi dengan Teknik - Teknik Heuristik."* Graha Ilmu, Yogyakarta. 2005.

Lin, Yu-Min, et al. "A Tabu Search Algorithm For Maximum Parsimony Phylogeny Inference," *European Journal of Operational Research*, halaman 1908 - 1917, 2007.

Pressman. Roger S. "*Software Engineering: A Practioner's Approach.*" 5th edition, ISBN 0073655783. McGrow-Hill, New York. 2000.

Putra, Yendrika. "*Aplikasi Penjadwalan Perkuliahan Menggunakan Algoritma Genetika,"* UIN Suska Riau, Pekanbaru. 2009.

Rachmawati, Heni. "*Analisis Penyelesaian Masalah Penjadwalan Kuliah Menggunakan Teknik Pewarnaan Graph Dengan Algoritma Koloni Lebah.*" [Online] available at http://digilib.its.ac.id/public/ITS-Master-18059-2209206810-Paper.pdf, diakses tanggal 6 Juni 2012.

Rusdiyanto. "*Pemecahan Masalah Penjadwalan Kuliah Dengan Menggunakan Algoritma Max-Min Ant System.*" [Online] available at http://elib.unikom.ac.id/files/disk1/55/jbptunikompp-gdl-s1-2006-rusdiyanto-2742-resume.pdf, diakses tanggal 31 Oktober 2011.

Santosa, Budi. "*Tutorial Particle Swarm Optimization*", [online] available at http://www.ie.its.ac.id/downloads/publikasi/132085804_pso_budi.pdf, diakses tanggal 31 Oktober 2011.

Santosa, Budi, dan Paul Willy. "*Metoda Metaheuristik Konsep dan Implementasi.*" Guna Widya, Surabaya. 2011.

Schaerf, A. "*A Survey of Automated Timetabling*." Artificial Intelligence Review 13: halaman 87 - 127, 1999.

Setemen, Komang, dan Mauridhi Hery Purnomo. "Kombinasi Algoritma Genetika dan Tabu Search dalam Pembuatan Tabel Jadwal Mata Kuliah." *Seminar on Intelligent Technology and Its Applications* 2008, ISBN 978-979-8897-24 - 5, halaman 375 - 378, 2008.

Siregar, Darto Paulus. "*Optimasi Penjadwalan Kuliah dengan Metode Tabu Search.*" [Online] available at http://repository.usu.ac.id/handle/123456789/21792, diakses tanggal 31 Oktober 2011.

Solimanpur, M., et al. "A Heuristic to Minimize Makespan of Cell Scheduling Problem." *Int. J. Production Economics*, halaman 231 - 241, 2004.

Suyanto. "*Artificial Intelligence, Searching, Reasoning, Planning dan Learning.*" Informatika, Bandung. 2007.

Suyanto. "*Evolutionary Computation.*" Informatika, Bandung. 2008.

UIN Suska. "*Buku Panduan dan Informasi Akademik Tahun Akademik 2006/2007*". UIN Suska Press, Pekanbaru. 2006.

Zerda, Evi Ria. "*Analisis dan Penerapan Algoritma Particle Swarm Optimization (PSO) pada Optimasi Penjadwalan Sumber Daya Proyek.*" Institut Teknologi Telkom, Bandung. 2009.