# PENERAPAN METODE *SIMULATED ANNEALING* UNTUK PENJADWALAN PERKULIAHAN

Muhammad Lukman Hakim[1], Makmur Hasibuan[2]
Program Studi Sistem Informasi STIKOM Medan
Email : [1]sixteen.mey@gmail.com, [2]makmur281077@gmail.com

***Abstrak***

*Pembuatan jadwal perkuliahan suatu perguruan tinggi sering terkendala karena keterbataan sarana dan prasarana berupa: (1) Sedikitnya dosen tetap yang ada; (2) Ketersediaan slot waktu dosen freelance yang sangat sempit dan (3) Jumlah dan kapasitas ruangan belajar (teori/praktek) yang minim. Sebuah studi kasus optimasi penjadwalan perkuliahan telah dilakukan pada lembaga pendidikan AMIK INTeL Com GLOBAL INDO yang berkedudukan di kota Kisaran. Lembaga pendidikan tersebut telah memiliki seperangkat software berbasis web yang dibuat dengan bahasa pemrograman PHP dibantu database MySQL untuk melakukan penjadwalan perkuliahan, akan tetapi sistem pengalokasian kelas mengajarnya belum terotomatisasi. Sistem yang ada sudah bisa mengantisipasi terjadinya bentrok dalam penyusunan jadwal perkuliahan dengan penerapan query MySQL, akan tetapi peneliti menganggap bahwa sistem yang ada belum bisa mengoptimalkan pengalokasian kelas mengajar dan dosen pengampu kepada ruangan belajar dan waktu. Ada banyak metode penjadwalan (scheduling) yang sudah pernah dibahas oleh peneliti sebelumnya antara lain: Particle Swarm Optimization (PSO) (Sivanandam, et al., 2007), Algoritma Genetika (Harsora dan Shah, 2011), dan Simulated Annealing (Tamilarasi dan Kumar, 2010). Peneliti memilih metode Simulated Annealing karena memiliki keunggulan dari sisi kecepatan dibandingkan dengan metode-metode lainnya. Di samping itu peneliti ingin memodifikasi mekanisme Simulated Annealing yang telah dibuat oleh peneliti sebelumnya sehingga didapatkan kualitas penjadwalan mata kuliah yang lebih optimal.*

Kata Kunci : *Simulated Annealing, Constraint, Scheduling,* PHP dan MySQL

***Abstarct***

*Making a lecture schedule for a university is often constrained due to limited facilities and infrastructure in the form of: (1) At least there are permanent lecturers; (2) The availability of free lance lecturer time slots is very narrow and (3) The number and capacity of study rooms (theory/practice) are minimal. A case study of optimizing lecture scheduling has been carried out at the AMIK INTeL Com GLOBAL INDO educational institution based in the city of Kisaran. The educational institution already has a set of web-based software made with the PHP programming language assisted by a MySQL database to schedule lectures, but the teaching class allocation system has not been automated. The existing system has been able to anticipate the occurrence of clashes in the preparation of the lecture schedule with the application of MySQL queries, but the researcher considers that the existing system has not been able to optimize the allocation of teaching classes and lecturers to study rooms and time. There are many scheduling methods that have been discussed by previous researchers, including: Particle Swarm Optimization (PSO) (Sivanandam, et al., 2007), Genetic Algorithm (Harsora and Shah, 2011), and Simulated Annealing (Tamilarasi and Kumar , 2010). The researcher chose the Simulated Annealing method because it has advantages in terms of speed compared to other methods. In addition, researchers want to modify the Simulated Annealing mechanism that has been made by previous researchers so that a more optimal quality of course scheduling is obtained.*

*Keywords: Simulated Annealing, Constraint, Scheduling,* PHP and MySQL

## 1. PENDAHULUAN

Adapun permasalahan sistem penjad-walan perkuliahan pada lembaga pendidikan AMIK INTeL Com GLOBAL INDO Kisaran sesuai dengan hasil pengamatan peneliti adalah sebagai berikut:

a. Sulitnya bagi pihak administrasi untuk mengalokasikan ruang dan waktu sebuah kelas mengajar sesuai dengan kesediaan dosen pengampu karena banyaknya kemungkinan yang ada dan adanya kendala dalam sarana dan prasarana seperti yang telah diuraikan sebelumnya.
b. Jadwal perkuliahan yang dihasilkan oleh sistem yang ada belum optimal dalam arti kata pengalokasian kelas mengajar dengan dosen pengampu kepada ruangan belajar dan waktu belum tersusun secara efektif dan efisien. Hal ini dilihat dengan masih terjadinya *fragmentasi* pada

pemakaian ruangan maupun pengalokasian dosen yaitu pada suatu slot tertentu masih ada ruangan kosong sementara di sisi lain dosen belum teralokasi sesuai dengan kesediaanya.

Sesuai dengan permasalahan yang telah dirumuskan di atas, maka peneliti menganggap perlu membatasi ruang lingkup masalah dalam menghasilkan jadwal perkuliahan yang optimal, yaitu:

a. Sebuah jadwal dianggap optimal apabila sudah benar-benar bebas dari bentrok baik bentrok dosen, bentrok kelas mahasiswa, bentrok ruangan maupun bentrok dengan sholat Jum'at dan apakah kapasitas ruangan yang dialokasikan sudah sesuai dengan jumlah anggota kelas mahasiswa. Dengan kata lain jadwal sudah memenuhi aturan penjadwalan (*constraint*) yang sudah ditentukan sebelum-nya.
b. Sebuah jadwal dikatakan optimal jika pengalokasian waktu bagi setiap kelas mengajar yang ada sudah sesuai dengan kesediaan hari dan slot waktu dosen yang bersangkutan.
c. Sebuah jadwal dikatakan optimal jika tidak terdapat fragmentasi dalam pengalokasian ruang dan waktu. Fragmentasi yang dimaksudkan adalah terdapatnya ruangan kosong pada slot waktu tertentu sementara dosen belum teralokasi sesuai dengan kesediaannya.

## 2. METODE PENELITIAN

### 2.1. Penjadwalan Perkuliahan

Penjadwalan adalah pengaturan atau penyusunan kegiatan berdasarkan waktu yang ada agar dalam pelaksanaan kegiatan bisa berjalan teratur dan tidak saling berbenturan antara satu kegatan dengan kegiatan lainnya. Dalam pembuatannya harus memperhatikan beberapa hal seperti waktu yang tersedia, jumlah tempat pelaksanaan, banyaknya kegiatan, banyaknya peserta kegiatan dan banyaknya pelaksana kegiatan.

Penjadwalan terkonsentrasi kepada penanganan pengalokasian sumber daya terbatas untuk dioptimasi menggunakan sebuah penerapan kriteria yang terdiri dari pemakaian waktu atau penggunaan biaya (*cost*)[1].

Masalah penjadwalan dapat dikategorikan kepada klass masalah *NP-Complete* (*NPC*)[2]. *NP* adalah singkatan dari *Non-deterministic Polynomial time* yang merupakan suatu klass masalah komputa-sional yang kompleks (*Computational Complexity*) untuk masalah pengambilan keputusan (*Decision Problem*).

### 2.2. *Simulated Annealing*

Algoritma *Simulated Annealing* (SA) adalah suatu teknik yang tangguh untuk menyelesaikan masalah optimisasi kombinatorial yang berat tanpa struktur yang khusus. Metode ini memiliki suatu kemampuan untuk melepaskan diri *local minima* dengan melakukan pergerakan bebas. Keuntungan utama dari algoritma SA adalah tidak membutuhkan memori komputer yang besar. Algorima SA didasari oleh metode iteratif yang diperkenalkan oleh Metropolis pada tahun 1953 yang mensimulasikan pergerakan atom-atom pada titik equilibrium yang diberikan oleh suhu. Kelemahan utama dari metode ini adalah membutuhkan waktu yang sangat lama untuk melakukan proses komputasi juga membutuhkan jumlah iterasi yang sangat besar untuk menemukan kekonvergensian[3]

*Simulated Annealing* didasari oleh sebuah teknik optimisasi stokastik tekhnik yang diinspirasi oleh the proses kristalisasi seperti proses pendinginan metal (logam)[4]. *Annealing* dalam ilmu metalurgi dan material dimulai dengan sebuah proses pemanasan dan diikuti oleh pendinginan terkontrol sebuah material untuk memperoleh kristal dengan kerugian/energi minimum.

*Simulated Annealing* berjalan berdasarkan analogi dengan proses *annealing* yang telah dijelaskan di atas. Pada awal proses *Simulated Annealing*, dipilih suatu solusi awal yang merepresentasikan kondisi materi sebelum proses dimulai. Gerakan bebas dari atom-atom pada materi, direpresentasikan dalam bentuk modifikasi terhadap solusi awal/solusi sementara. Pada awal proses *Simulated Annealing*, saat parameter suhu (t) diatur tinggi, solusi sementara yang sudah ada diperbolehkan untuk mengalami modifikasi secara bebas. Kebebasan ini secara relatif diukur berdasarkan nilai fungsi tertentu yang mengevaluasi seberapa optimal solusi sementara yang telah diperoleh. Bila nilai fungsi evaluasi hasil modifikasi ini membaik (dalam masalah optimisasi yang berusaha mencari minimum berarti nilainya lebih kecil/*downhill*) solusi hasil modifikasi ini akan digunakan sebagai solusi selanjutnya. Bila nilai fungsi evaluasi hasil modifikasi ini memburuk, pada

saat temperatur *annealing* masih tinggi, solusi yang lebih buruk (*uphill*) ini masih mungkin diterima dengan melihat probabilitas penambahan energi *(δE),* yang diberikan oleh rumus :

$$P(\delta E) = exp(-\delta E /kt) \quad (1)$$

Di mana *k* adalah sebuah konstanta yang dikenal dengan konstanta Boltzmann.
Nilai *P(δE)* kemudian dibandingkan dengan suatu bilangan random antara 0 dan 1. Jika nilai *P(δE)* ternyata lebih besar, maka solusi diterima, sebaliknya solusi ditolak.

Dalam tahapan selanjutnya saat temperatur sedikit demi sedikit dikurangi, maka kemungkinan untuk menerima langkah modifikasi yang tidak memperbaiki nilai fungsi evaluasi semakin berkurang, sehingga kebebasan untuk memodifikasi solusi semakin menyempit, sampai akhirnya diharapkan diperoleh solusi yang mendekati solusi optimal.

Berikut ini adalah pemetaan dari *Physical Annealing* ke *Simulated Annealing* [5].

**Table 2.1. Pemetaan *Physical Annealing* ke *Simulated Annealing***

| ***Physical Annealing* (termodinamika)** | ***Simulated Annealing*** |
|---|---|
| Keadaan sistem | Solusi yang mungkin |
| Energi | Biaya |
| Perubahan keadaan | Solusi tetangga |
| Temperatur | Parameter kontrol |
| Keadaan beku | Solusi heuristik |

### 2.2.1. Algoritma *Simulated Annealing*

Struktur algoritma *Simulated Annealing* secara umum adalah sebagai berikut[6]:

1). Inisialisasi.
   a. Set iterasi counter $i = 1$.
   b. Hasilkan sebuah solusi awal *S* yang layak dan pastikan *S* adalah sebagai solusi optimal.
   c. Set temperature awal $T_i$ dan temperature akhir $T_f$.
   d. Definisikan sebuah fungsi pendingin $D(Ti)$.

2). Menghasilkan sebuah solusi tetangga yang layak.
   Terapkan fungsi tetangga kepada solusi *S* yang ada untuk memperoleh sebuah solusi tetangga yang baru *S'*.

3). Mengevaluasi Solusi Aktif dengan Solusi Tetangga.
   a. Jika nilai fungsi objektif dari solusi baru *S'* tidak lebih kecil dari fungsi objektif dari solusi aktif yaitu $\Phi(S') \geq \Phi(S)$ proses langkah 4.
   b. Jika $\Phi(S') < \Phi(S)$ lakukan $S = S'$, kemudian proses langkah 5.

4). Menguji kondisi Metropolis.
   a. Tentukan perbedaan $\Delta C$ antara Solusi terpilih *S* and Solusi tetangga *S'* sebagai $\Delta\Phi = \Phi(\mathbf{S'}) - \Phi(\mathbf{S})$, kemudian bangkitkan bilangan random $\rho$ diantara 0 dan 1.
   a. Jika $\rho < \exp(-\Delta\Phi/Ti)$ lakukan $\mathbf{S} = \mathbf{S'}$ kemudian proses langkah 5.

5). Cek penambahan kounter.
   b. Set $i \leftarrow i + 1$.
   c. Jika $i \leq N$ kembali ke langkah 2, jika tidak proses langkah 6.

6). Menyesuaikan temperatur.
   Sesuaikan temperatur menggunakan fungsi pendingin secara matematis yaitu dengan rumus $Ti \leftarrow D(Ti)$.

7). Cek konvergensi.
   Jika $Ti \geq Tf$ maka reset nilai $i = 1$ dan kembali ke langkah 2.
   Jika tidak berhenti dan tampilkan solusi optimal *S.*

### 2.2.2. *Annealing Schedule*

Ada 4 (empat) hal yang harus diperhatikan dalam proses *annealing* yaitu : dalam menentukan suhu awal, suhu akhir, pengurangan suhu dan jumlah iterasi pada setiap suhu.

**a Suhu Awal**

Suhu harus di-*setting* cukup tinggi untuk memberi "pergerakan" bebas untuk mendapatkan solusi tetangga yang berhampiran, akan tetapi jangan di-*setting* terlalu tinggi karena akan memakan

periode waktu yang lama dalam melakukan pencarian acak solusi tetangga. Pengalaman dan pemahaman terhadap masalah akan menemukan solusi yang tepat dalam pemilihan suhu awal.

Jika kita tahu maksimum perubahan yang akan terjadi dalam fungsi penghitungan cost, kita dapat menggunakannya untuk mengestimasi suhu awal. Mulailah dengan suhu tinggi, kurangi suhu dengan cepat sampai kira-kira 60% pergerakan buruk diterima. Gunakanlah ini sebagai suhu awal.

**b. Suhu Akhir**

Dalam proses *annealing b*iasanya suhu akan diturunkan sampai mencapai 0. Pemilihan proses penurunan suhu yang kurang tepat akan membuat algoritma berjalan sangat lama terutama jika menggunakan *geometric cooling schedule*. Dalam prakteknya tidak perlu untuk membiarkan suhu mencapai 0 karena peluang untuk menerima pergerakan buruk adalah hampir sama dengan menjadikan suhu menjadi sama dengan 0. Oleh karena itu kriteria penghentian algoritma dapat juga dicocokkan dengan suhu rendah atau ketika sistem sudah "beku" pada suhu yang bersangkutan (contohnya: tidak ada lagi pergerakan buruk yang bisa diterima).

**c. Penurunan Suhu**

Teori *Simulated Annealing* menyatakan bahwa kita harus memperkenankan iterasi yang cukup pada setiap suhu sehingga sistem stabil pada suhu tersebut. Sayangnya, teori ini juga menyatakan bahwa jumlah iterasi pada setiap suhu untuk mencapai kestabilan system juga meningkat secara eksponensial sesuai dengan ukuran masalah. Kita dapat juga melakukannya dengan jumlah iterasi yang besar dengan suhu yang kecil, angka iterasi yang kecil untuk beberapa suhu atau dengan menyeimbangkan diantara kedua cara tersebut.

Ada 2 (dua) metode penurunan suhu yang digunakan, yaitu:

1) Penurunan secara *linear* diberikan oleh rumus:
   $temp = temp - \alpha$
2) Penurunan secara *geometric* diberikan oleh rumus:
   $temp = temp * \alpha$

Pengalaman menunjukkan bahwa nilai $\alpha$ secara *geometric* diantara 0.8 dan 0.99 akan menemukan hasil yang lebih baik pada akhirnya. Tentu saja, nilai α yang tinggi akan memakan waktu yang lama untuk penghentian algoritma.

**d. Jumlah Iterasi pada Setiap Suhu**

Untuk menentukan jumlah iterasi pada setiap suhu kita bisa saja memberikan nilai iterasi yang tetap untuk setiap suhu. Metode lain adalah yang disarankan oleh Lundy, (1986) adalah dengan hanya melakukan satu iterasi pada setiap temperatur tetapi dengan menurunkan suhu secara perlahan dengan rumus penurunan suhu sebagai berikut:

$$t = t/(1 + \beta t) \quad (2)$$

Di mana β adalah merupakan nilai yang sangat kecil (misalnya: 0.001)

Alternatif lain adalah secara dinamis mengubah banyaknya iterasi sesuai dengan proses algoritma. Pada suhu rendah sangat penting untuk memberikan jumlah iterasi yang besar agar nilai *local optimum* dapat sepenuhnya dieksplorasi sedangkan pada suhu tinggi jumlah iterasi dapat dibuat kecil.

## 3. HASIL DAN PEMBAHASAN

Meskipun *annealing schedule* sudah mencakup keseluruhan *Simulated Annealing* itu sendiri, akan tetapi ada lagi pengambilan keputusan lain yang perlu dibuat berkenaan dengan masalah *Simulated Annealing* yaitu membuat fungsi evaluasi dan pembangkitan solusi tetangga.

**a. Pembuatan Fungsi Evaluasi (Fungsi Cost)**

Sebagaimana yang telah dijelaskan sebelumnya bahwa fungsi evaluasi adalah untuk mengecek apakah suatu solusi tetangga yang dibangkitkan (*S'*) layak atau tidak layak diterima. Fungsi evaluasi dilakukan dengan cara menghitung nilai kesalahan yang terjadi pada solusi tetangga (*S'*) pada setiap iterasi algoritma.

Seringkali pembuatan fungsi evaluasi ini merupakan bagian tersulit dari pembuatan suatu algoritma *Simulated Annealing*, oleh karena itu kita harus mengevaluasi fungsi cost seefisien mungkin. Fungsi evaluasi dapat dilakukan pada sebagian kecil solusi tetangga *(S')* yang disebut

dengan *Delta Evaluation* atau sebagian besar dari solusi tetangga *(S')* yang disebut dengan *Partial Evaluation.*

Jika memungkinkan fungsi evaluasi dapat juga didisain sehingga memper-panjang proses pencarian. Satu cara yang harus dipenuhi adalah menghindari fungsi evaluasi yang mengembalikan nilai yang sama pada beberapa keadaan. Ini dapat dilihat sebagai representasi dari sebuah "dataran tinggi" pada ranah pencarian. Dengan kata lain proses pencarian tidak memiliki pengetahuan jalan mana yang harus dilanjutkan.

Pada kenyataannya ketika algoritma *Simulated Annealing* berjalan iterasi demi iterasi, fungsi evaluasi akan menemui solusi ilegal yang tidak diperkenankan, untuk itu ada 2 (dua) *constraint* yang akan diperhitungkan, yaitu:

1) *Hard Constraints*: *constraint* ini tidak boleh bentrok pada solusi yang layak.
2) *Soft Constraints*: *constraint* ini, idealnya tidak boleh bentrok, tetapi jika bentrok, solusi tetap layak.

*Hard constraints* diberikan bobot yang besar. Solusi yang memiliki bentrok pada *constraint* ini akan memiliki fungsi cost yang tinggi. *Soft constraints* diberi bobot tergantung tingkat kepentingannya. Pembobotan dapat secara dinamis diubah sesuai dengan proses algoritma. Pelanggaran terhadap *hard constraints* dapat diterima pada awal algoritma tetapi kemudian ditolak untuk proses selanjutnya.

**b. Pembangkitan Solusi Tetangga (*Neighbour-hood*)**

Ada 2 (dua) masalah utama dalam pembangkitan solusi tetangga dalam spefisikasi masalah khusus, yaitu:

1) Bagaimanakah kita melakukan "pergerakan" dari suatu keadaan ke keadaan lainnya ?.
2) Ketika kita berada pada suatu keadaan tertentu dalam suatu algoritma, apakah keadaan lain yang mesti dicapai?

Beberapa hasil pengujian telah menunjukkan bahwa struktur tetangga haruslah simetris, artinya jika kita bergerak dari keadaan *i* ke keadaan *j* maka harus dimungkinkan untuk bergerak dari keadaan *j* ke keadaan *i,* akan tetapi untuk kondisi yang jarang tidak harus simetris sesuai dengan tingkat keyakinan konvergensi. Harus dimungkinkan untuk bergerak dari suatu keadaan ke keadaan lainnya, oleh karena itu yang terpenting adalah kita dapat menerapkan hal tersebut terhadap masalah yang akan diselesaikan.

### 3.1. Analisa Sistem yang Sedang Berjalan

AMIK INTeL Com GLOBAL INDO memiliki kegiatan belajar mengajar yang dilaksanakan pada hari Senin sampai Sabtu untuk kelas pagi yang dimulai pukul 07.30 sampai dengan pukul 17.15 WIB dan hari Senin sampai Jum'at untuk kelas malam yang dimulai pukul 17.30 sampai dengan pukul 22.00 WIB.

**a. Komponen Pembentuk Jadwal Perkuliahan**

Ada 4 (empat) komponen pembentuk jadwal perkuliahan yaitu: satuan waktu kuliah, dosen beserta mata kuliah yang diampunya, kelas mahasiswa dan ruangan.

1) Satuan Waktu Kuliah

Komponen utama yang memiliki peranan penting dalam membentuk jadwal perkuliahan adalah satuan waktu (durasi) yang dibutuhkan untuk 1 (satu) Sistem Kredit Semester (SKS). Adapun durasi yang ditetapkan oleh AMIK INTeL Com GLOBAL INDO Kisaran untuk Sistem Kredit Semester adalah 45 (empat puluh lima) menit untuk setiap SKS.

2) Dosen dan Mata kuliah

Komponen pembentuk jadwal perkuliahan selanjutnya adalah dosen dan mata kuliah. Pemberian tugas mengajar suatu mata kuliah kepada seorang dosen disebut dengan pengampuan. Sebagai contoh pada gambar 4.1. dapat dilihat bahwa dosen dengan kode SDIMN mengampu tiga mata kuliah yaitu Akuntansi Biaya II, Akuntansi Keuangan Lanjutan II dan Akuntansi Keuangan Menengah dan Praktek.

3) Kelas Mahasiswa

Komponen pembentuk jadwal perkuliahan selanjutnya adalah kelas mahasiswa sebagai peserta perkuliahan. Pada lembaga pendidikan AMIK INTeL Com GLOBAL INDO Kisaran pengambilan mata kuliah oleh mahasiswa adalah bersifat paket dengan jumlah mata kuliah yang sama untuk setiap mahasiswa. Dengan kata lain seorang mahasiswa tidak diperkenankan untuk

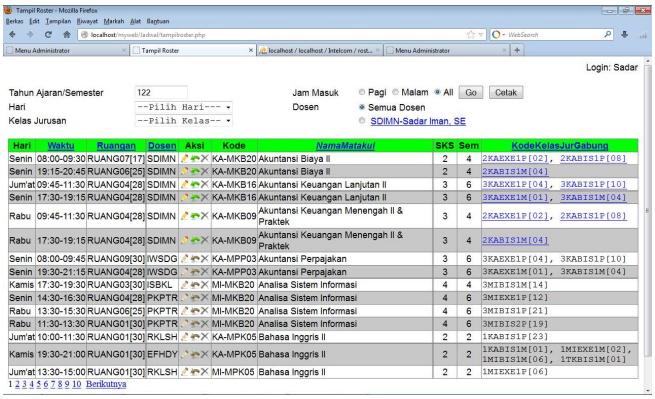
mengulang mata kuliah pada semester sebelumnya atau mengambil mata kuliah pada semester berikutnya. Di samping itu seorang mahasiswa yang terdaftar pada kelas pagi tidak bisa kuliah pada malam hari atau sebaliknya.

4) Ruangan

Komponen pembentuk jadwal perkuliahan terakhir adalah ruangan sebagai tempat untuk menyelenggarakan perkuliahan yang juga merupakan tempat bertemunya komponen-komponen sebelumnya yaitu satuan waktu, dosen beserta mata kuliah dan kelas mahasiswa.

**b. Penyusunan Jadwal Perkuliahan**

Proses penyusunan jadwal perkuliahan AMIK INTeL Com GLOBAL INDO Kisaran diawali dengan penawaran mata kuliah pada tiap semester kepada dosen yang kompeten pada mata kuliah tersebut. Dosen dapat memilih mata kuliah yang ditawarkan dengan cara memilih mata kuliah tersebut dengan mengklik gambar ikon pensil di kolom Aksi pada antar muka yang telah disediakan seperti terlihat pada gambar 1 berikut ini.




| Hari | Waktu | Ruangan | Dosen | Aksi | Kode | NamaMatakul | SKS | Sem | KodeKelasJurGabung |
|---|---|---|---|---|---|---|---|---|---|
| Senin | 08:00-09:30 | RUANG07[17] | SDIMN | | KA-MKB20 | Akuntansi Biaya II | 2 | 4 | 2KAEXE1P[02], 2KABIS1P[08] |
| Senin | 19:15-20:45 | RUANG06[25] | SDIMN | | KA-MKB20 | Akuntansi Biaya II | 2 | 4 | 2KABIS1M[04] |
| Jum'at | 09:45-11:30 | RUANG04[28] | SDIMN | | KA-MKB16 | Akuntansi Keuangan Lanjutan II | 3 | 6 | 3KAEXE1P[04], 3KABIS1P[10] |
| Senin | 17:30-19:15 | RUANG04[28] | SDIMN | | KA-MKB16 | Akuntansi Keuangan Lanjutan II | 3 | 6 | 3KAEXE1M[01], 3KABIS1M[04] |
| Rabu | 09:45-11:30 | RUANG04[28] | SDIMN | | KA-MKB09 | Akuntansi Keuangan Menengah II & Praktek | 3 | 4 | 2KAEXE1P[02], 2KABIS1P[08] |
| Rabu | 17:30-19:15 | RUANG04[28] | SDIMN | | KA-MKB09 | Akuntansi Keuangan Menengah II & Praktek | 3 | 4 | 2KABIS1M[04] |
| Senin | 08:00-09:45 | RUANG09[30] | WSDG | | KA-MPP03 | Akuntansi Perpajakan | 3 | 6 | 3KAEXE1P[04], 3KABIS1P[10] |
| Senin | 19:30-21:15 | RUANG04[28] | WSDG | | KA-MPP03 | Akuntansi Perpajakan | 3 | 6 | 3KAEXE1M[01], 3KABIS1M[04] |
| Kamis | 17:30-19:30 | RUANG03[30] | ISBKL | | MI-MKB20 | Analisa Sistem Informasi | 4 | 4 | 3MIBIS1M[14] |
| Senin | 14:30-16:30 | RUANG04[28] | PKPTR | | MI-MKB20 | Analisa Sistem Informasi | 4 | 6 | 3MIEXE1P[12] |
| Rabu | 13:30-15:30 | RUANG06[25] | PKPTR | | MI-MKB20 | Analisa Sistem Informasi | 4 | 6 | 3MIBIS1P[21] |
| Rabu | 11:30-13:30 | RUANG01[30] | PKPTR | | MI-MKB20 | Analisa Sistem Informasi | 4 | 6 | 3MIBIS2P[19] |
| Jum'at | 10:00-11:30 | RUANG01[30] | RKLSH | | KA-MPK05 | Bahasa Inggris II | 2 | 2 | 1KABIS1P[23] |
| Kamis | 19:30-21:00 | RUANG01[30] | EFHDY | | KA-MPK05 | Bahasa Inggris II | 2 | 2 | 1KABIS1M[01], 1MIEXE1M[02], 1MIBIS1M[06], 1TKBIS1M[01] |
| Jum'at | 13:30-15:00 | RUANG01[30] | RKLSH | | MI-MPK05 | Bahasa Inggris II | 2 | 2 | 1MIEXE1P[06] |

1 2 3 4 5 6 7 8 9 10 Berikutnya

**Gambar 1. Antar Muka Penjadwalan Perkuliahan**

**c. Aturan Penjadwalan (*Constraint*)**

Aturan penjadwalan yang berlaku pada lembaga pendidikan AMIK INTeL Com GLOBAL INDO Kisaran dibagi menjadi 2 (dua) aturan penjadwalan yaitu *Hard Constraint* dan *Soft Constrain*. *Hard Constrain* didefinisikan sebagai berikut:

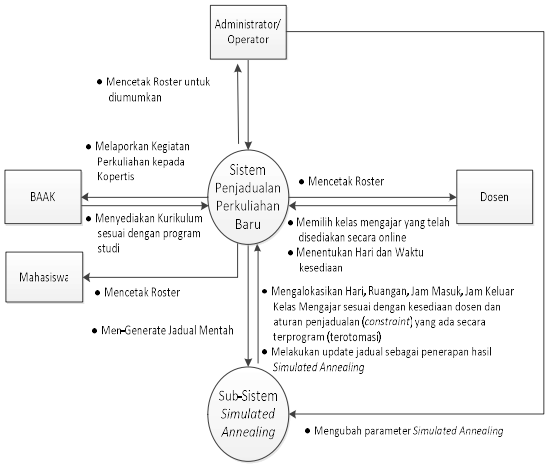
1) Seorang dosen hanya bisa mengajar satu kelas perkuliahan pada suatu waktu. Tidak dimungkinkan seorang dosen mengajar lebih dari satu kelas perkuliahan pada waktu yang bersamaan.
2) Satu kelas mahasiswa hanya bisa mengikuti satu kelas perkuliahan pada satu waktu. Tidak dimungkinkan satu kelas mahasiswa mengikuti lebih dari satu kelas perkuliahan pada waktu yang bersamaan.
3) Sebuah ruangan hanya bisa dipakai oleh satu kelas perkuliahan pada satu waktu. Tidak dimungkinkan satu ruangan dipakai oleh lebih dari satu kelas perkuliahan pada waktu yang bersamaan.
4) Sebelum jadwal perkuliahan dibuat, suatu mata kuliah ditawarkan hanya jika ada dosen yang bisa mengampunya.
5) Satu jenis mata kuliah dapat diampu oleh lebih dari seorang dosen.
6) Seorang dosen bisa mengampu lebih dari satu mata kuliah akan tetapi maksimum hanya 3 (tiga) jenis mata kuliah yang boleh diampu.
7) Waktu kuliah kelas pagi dimulai dari pukul 07.30 sampai dengan pukul 17.30 WIB dan waktu kuliah kelas malam dimulai dari pukul 17.30 sampai dengan pukul 22.00 WIB.
8) Satuan waktu kuliah adalah 45 (empat puluh lima) menit per SKS.
9) Dalam jangka waktu satu hari terdapat 13 slot waktu untuk kelas pagi dan 6 slot waktu untuk kelas malam sehingga dalam siklus 1 (satu) pekan (6 hari untuk kelas pagi dan 5 hari untuk kelas malam) terdapat 109 slot waktu.
10) Untuk hari Jum'at pada les ke-7 dan ke-8 (12.00 s.d. 13.30) tidak ada perkuliahan karena bersamaan dengan waktu pelaksanaan Shalat Jum'at.

Sedangkan *Soft Constraint* didefinisikan sebagai berikut:

1) Sebuah kelas mahasiswa tidak boleh mengikuti lebih dari 2 kelas belajar per hari.
2) Seorang dosen tidak boleh mengajar lebih dari 3 kelas mengajar per hari.

### 3.2. Penerapan *Simulated Annealing* untuk Optimasi Penjadwalan Perkuliahan Context Diagram

Untuk menerapkan metode *Simulated Annealing* dalam melakukan Optimasi Penjadwalan Perkuliahan pada AMIK INTeL Com GLOBAL INDO Kisaran, maka peneliti akan menjelaskan system yang sedang berjalan pada saat ini sebagaimana administrator/operator melakukan pengalokasian Hari, Jam Masuk, Jam Keluar dan Ruangan terhadap jadwal mentah (kelas mengajar) yang dihasilkan oleh sistem. Akan tetapi, karena masih mengandalkan pemikiran manusia, jadwal yang dihasilkan belum optimal dan proses penyelesaiannya masih membutuhkan waktu. Sesuai dengan pengalaman peneliti yang bekerja sebagai dosen sekaligus administrator pada lembaga pendidikan AMIK INTeL Com GLOBAL INDO Kisaran, penyelesaian sekitar 200-an jadwal perkuliahan membutuhkan waktu sekitar 2 (dua) sampai 3 (tiga) hari. Untuk itu peneliti ingin melakukan perubahan terhadap sistem yang sudah ada yaitu dengan membuat sebuah sub-sistem *Simulated Annealing* dalam pengalokasian Hari, Jam Masuk, Jam Keluar dan Ruangan terhadap semua jadwal mentah yang ada sehingga jadwal perkuliahan yang dihasilkan nantinya diharapkan bisa lebih optimal. Untuk lebih jelasnya dapat dilihat pada gambar 3 berikut.




**Gambar 2. Context Diagram Sistem Penjadwalan Perkuliahan Baru yang Diusulkan**

### 3.3. Analisa Kebutuhan Sistem

Beranjak dari objek permasalahan yang akan diteliti yaitu berupa optimasi penjadwalan perkuliahan, maka peneliti mencoba menganalisa kebutuhan apa saja yang mesti dipenuhi untuk menyelesaikan masalah. Secara garis besar kebutuhan penelitian dapat dibagi kepada 2 (dua) bagian yaitu:

**a. Analisa Input**

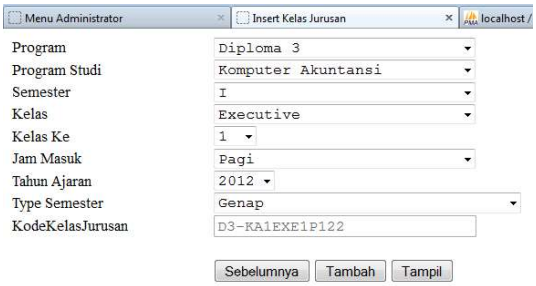
Data yang dibutuhkan untuk pembuatan jadwal perkuliahan pada sistem yang sedang berjalan ada 5 (lima) buah entitas, yaitu :

a. Data dosen.
b. Data kelas mahasiswa.
c. Data mata kuliah.
d. Data ruangan.
e. Data jadwal mentah perkuliahan yang berisikan daftar berpasangan antara mata kuliah yang akan dibawakan oleh seorang dosen dengan kelas mahasiswa.

Data input untuk dosen terdiri dari kode dosen, nama dosen, jenis kelamin, alamat, No. HP, kepangkatan, pendidikan terakhir, alumni, status masih aktif, nama user dan password. Kode dosen lebarnya 5 (lima) digit adalah merupakan singkatan nama dosen yang ditentukan oleh pihak administrasi ketika data seorang dosen diinput pertama kali. Kepangkatan seorang dosen berguna

untuk menentukan gaji atau honor per SKS mata kuliah yang akan diampunya. Nama user dan password untuk masuk ke sistem dapat diganti sendiri oleh dosen yang bersangkutan.

Data kelas mahasiswa adalah gabungan mahasiswa yang memiliki angkatan (tahun masuk) dan jurusan yang sama. Penginputan data ini hanya dilakukan sekali saja untuk setiap kelas yaitu pada semester I, sedangkan untuk semester-semester berikutnya nilai field semester akan naik terus disesuaikan oleh sistem. Untuk lebih jelasnya dapat dilihat pada gambar 3 berikut:




**Gambar 3. Form Input Data Kelas Mahasiswa**

Dari gambar 3 di atas dapat dilihat bahwa form tersebut memiliki 9 (sembilan) buah property yaitu program, program studi, semester, kelas, kelas ke, jam masuk, tahun ajaran, tipe semester, dan kode kelas jurusan. Program berisi 2 (dua) pilihan yaitu Diploma 3 dan Strata 1. Program studi berisi 3 (tiga) pilihan yaitu Komputer Akuntansi, Manajemen Informatika dan Teknik Komputer. Semester berisi 6 (enam) pilihan yaitu I, II, III sampai dengan VI. Kelas berisi 2 (dua) pilihan yaitu Executive dan Bisnis. Kelas Ke berisi 10 (sepuluh) pilihan yaitu 1,2,3 sampai dengan 10. Jam masuk berisi 2 (dua) pilihan yaitu Pagi dan Malam. Tahun Ajaran berisi 3 (tiga) pilihan yaitu 1 (satu) tahun sebelum Tahun Ajaran Aktif, Tahun Ajaran Aktif dan 1 (satu) tahun setelah Tahun Ajaran Aktif. Jika Tahun Ajaran Aktif adalah 2012, maka pilihan Tahun Ajaran adalah 2011, 2012 dan 2013. Tipe Semester berisi 2 (dua) pilihan yaitu Ganjil dan Genap. *KodeKelasJurusan* adalah pengkodean kelas mahasiswa yang merupakan perpaduan dari field-field sebelumnya.

Data mata kuliah yang ditawarkan pada setiap semester pada lembaga pendidikan adalah bersifat paket artinya semua mata kuliah yang ada wajib diambil oleh mahasiswa tanpa melihat Indeks Prestasi Kumulatif (IPK) semester sebelumnya.

**P**engkodean mata kuliah diawali dengan dua digit yang merupakan singkatan nama jurusan. Field *No*. berguna untuk menyusun mata kuliah berdasarkan urutan yang ditentukan oleh Ketua Program Studi yang bukan berdasarkan nama mata kuliah atau kode mata kuliah. Tipe mata kuliah berguna untuk mengalokasikan mata kuliah yang bersangkutan sesuai dengan tipenya. Jika tipenya Teori maka akan dialokasikan pada ruangan teori dan jika tipenya Praktek maka akan dialokasikan pada ruangan laboratorium.

Selanjutnya agar pengalokasian ruangan sesuai dengan tipe mata kuliah yang akan dialokasikan padanya, maka ruangan juga dibedakan menurut tipenya yaitu ruangan teori dan ruangan praktek. Kapasitas ruangan berguna untuk menyesuaikan kapasitas ruangan yang tersedia dengan banyaknya mahasiswa pada suatu kelas perkuliahan yang akan dialokasikan pada ruangan tersebut.

Pada setiap awal semester, sistem penjadwalan perkuliahan yang sudah ada akan menghasilkan suatu jadwal mentah yang didapatkan dari *query* antara kelas mahasiswa yang ada pada setiap semester digabungkan dengan mata kuliah yang ditawarkan pada semester yang bersangkutan sehingga menghasilkan daftar berpasangan antara mata kuliah dan kelas mahasiswa.

Semua daftar berpasangan kelas perkuliahan adalah bersifat *unique* artinya tidak boleh sama antara satu dengan yang lainnya. Untuk itu dibuatlah sebuah field *KodeKelasMengajar* yang nilainya merupakan gabungan antara kode mata kuliah ditambah 3 (tiga) digit pertama fungsi MD5 dari kode mata kuliah ditambah 2 (dua) digit terakhir tahun akademis ditambah 1 (satu) digit yang mewakili semester ganjil atau genap (1=ganjil, 2=genap).

**2. Analisa Output**

Output jadwal perkuliahan pada sistem yang sedang berjalan dapat dilihat pada tabel 1. Dari tabel tersebut bisa dilihat bahwa output terdiri dari sembilan kolom yaitu: *Hari, Waktu, Ruangan, Dosen, Kode, NamaMatakul, SKS, Sem, KodeKelasJurGabung.* Kolom *Hari* nantinya akan berisi hari dalam sepekan mulai dari Senin, Selasa sampai dengan Sabtu. Kolom *Waktu* akan berisi field *Jam Masuk* dan *Jam Keluar*. Kolom *Ruangan* akan berisi field *KodeRuangan* digabung dengan field *Kapasitas* yang berada didalam kurung siku. Kolom *Dosen* akan berisi field *KodeDosen.* Kolom *Kode* akan berisi field *KodeMatakul.* Kolom-kolom *NamaMatakul, SKS, Sem* dan *KodeKelasJurGabung* akan berisi data dengan nama field yang sama dengan judul kolomnya.

### 3.4. Perancangan Sistem

Sesuai dengan analisis kebutuhan sistem yang telah dilakukan sebelumnya untuk mengimplementasikan optimasi penjadwalan dengan metode *Simulated Annealing*, maka dalam tahap ini peneliti akan melakukan perancangan terhadap form input data, tabel-tabel database yang dibutuhkan serta perancangan program untuk menghasilkan ouput sebagaimana yang diuraikan sebelumnya.

**a. Disain Database**

Setelah peneliti melakukan analisa secara mendalam terhadap sistem penjadwalan perkuliahan yang sedang berjalan, maka peneliti menyim- pulkan harus menambah 4 (empat) buah tabel lagi yaitu tabel hari dalam sepekan, tabel slot waktu per hari, tabel slot waktu untuk per pekan dan tabel slot waktu kesediaan dosen pada database yang ada agar sistem penjadwalan yang sedang berjalan bisa mengadopsi penerapan metode *Simulated Annealing* yang diusulkan peneliti.

Tabel 1. Roster Perkuliahan AMIK INTeL Com GLOBAL INDO

| Hari | Waktu | Ruangan | Dosen | Kode | NamaMatakul | SKS | Sem | KodeKelasJurGabung |
|---|---|---|---|---|---|---|---|---|
| Senin | 08:00-09:30 | RUANG05[28] | RHMYN | MI-MPK03 | Pendidikan Kewarganegaraan | 2 | 2 | 1MIBIS3P[25] |
| Senin | 08:00-09:30 | RUANG02[30] | EVOSR | KA-MPB07 | Kewiraswastaan | 2 | 2 | 1KABIS1P[23] |
| Senin | 08:00-09:30 | L02-EXE[17] | SRRMD | TK-MKB03 | Komputer Aplikasi Teknik II (P.Point, MS-Access) | 2 | 2 | 1TKBIS2P[16] |
| Senin | 08:00-09:30 | RUANG07[17] | SDIMN | KA-MKB20 | Akuntansi Biaya II | 2 | 4 | 2KAEXE1P[02], 2KABIS1P[08] |
| Senin | 08:00-09:30 | RUANG01[30] | EFHDY | MI-MKP07 | Bahasa Inggris IV | 2 | 4 | 2MIBIS1P[23] |
| Senin | 08:00-09:45 | RUANG09[30] | IWSDG | KA-MPP03 | Akuntansi Perpajakan | 3 | 6 | 3KAEXE1P[04], 3KABIS1P[10] |
| Senin | 08:00-10:00 | L03-EXE[17] | SURJI | TK-MKB06 | Pemrograman Delphi | 4 | 4 | 2TKBIS1P[21] |
| Senin | 08:00-10:00 | L01-EXE[17] | JSSTP | MI-MKB09 | Disain Web | 4 | 4 | 2MIEXE1P[08] |
| Senin | 09:30-11:00 | RUANG01[30] | EVOSR | MI-MPB04 | Kewiraswastaan | 2 | 2 | 1MIBIS3P[25] |
| Senin | 09:30-11:00 | RUANG03[28] | MHISA | TK-MKB17 | Teori Rangkaian Listrik | 2 | 2 | 1TKBIS2P[16] |
| Senin | 09:30-11:00 | RUANG05[28] | RHMYN | MI-MPK03 | Pendidikan Kewarganegaraan | 2 | 2 | 1MIBIS2P[25] |
| Senin | 09:30-11:00 | RUANG02[30] | EFHDY | MI-MKP07 | Bahasa Inggris IV | 2 | 4 | 2MIBIS2P[25] |
| Senin | 09:30-11:00 | RUANG07[17] | SYAMS | TK-MPB06 | Kewiraswastaan | 2 | 6 | 3TKEXE1P[12], 3TKEXE2P[09] |
| Senin | 09:30-11:15 | L01-BIS[36] | SRRMD | KA-MKB02 | Paket Aplikasi II (P.Point,MS-Access) | 3 | 2 | 1KABIS1P[23] |
| Senin | 09:45-11:15 | RUANG09[30] | IWSDG | KA-MPB09 | Perpajakan | 2 | 4 | 2KAEXE1P[02], 2KABIS1P[08] |
| Senin | 10:00-12:00 | L01-EXE[17] | SURJI | TK-MKB06 | Pemrograman Delphi | 4 | 4 | 2TKEXE1P[09] |
| Senin | 10:00-12:00 | L02-EXE[17] | PRHSB | MI-MKB17 | Internet | 4 | 2 | 1MIBIS4P[20] |
| Senin | 11:00-12:30 | RUANG01[30] | RKLSH | MI-MPK05 | Bahasa Inggris II | 2 | 2 | 1MIBIS3P[25] |
| Senin | 11:00-12:30 | RUANG05[28] | SYAMS | TK-MPB06 | Kewiraswastaan | 2 | 6 | 3TKBIS1P[20] |
| Senin | 11:15-13:15 | L01-BIS[36] | JSSTP | MI-MKB09 | Disain Web | 4 | 4 | 2MIBIS1P[23] |
| Senin | 11:30-13:00 | RUANG09[30] | RHMYN | TK-MPK03 | Pendidikan Kewarganegaraan | 2 | 2 | 1TKBIS2P[16], 1MIEXE1P[06] |
| Senin | 12:00-13:45 | L02-EXE[17] | KHKLJ | KA-MKB07 | Paket Komputer Akuntansi II (MYOB II) | 3 | 4 | 2KAEXE1P[02], 2KABIS1P[08] |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Senin | 13:00-14:30 | L03-EXE[17] | SRRMD | MI-MKB23 | Pemrograman Orientasi Objek (OOP) I | 4 | 4 | 2MIEXE1P[08] |
| Senin | 13:30-15:00 | RUANG05[28] | RHMYN | KA-MPK03 | Pendidikan Kewarganegaraan | 2 | 2 | 1KABIS1P[23] |
| Senin | 14:30-16:00 | L02-EXE[17] | SLMSB | MI-MKB05 | Disain Animasi | 2 | 4 | 2MIEXE1P[08] |
| Senin | 14:30-16:30 | RUANG03[28] | PKPTR | MI-MKB20 | Analisa Sistem Informasi | 4 | 6 | 3MIEXE1P[12] |
| Senin | 15:00-16:30 | RUANG05[28] | RHMYN | TK-MPK03 | Pendidikan Kewarganegaraan | 2 | 2 | 1TKEXE1P[02], 1TKBIS1P[16] |
| Senin | 15:15-17:15 | L01-BIS[36] | JSSTP | MI-MKK06 | Pemrograman C++ | 4 | 2 | 1MIBIS2P[25] |

**Tabel 2. Hari dalam Sepekan**

| No. | Field | Tipe | Lebar | Keterangan |
|---|---|---|---|---|
| 1 | Hari | TINYINT | 1 | Primary Key |
| 2 | NamaHari | VARCHAR | 6 | Minggu, Senin, Selasa dst |

Tabel 2 ini berfungsi untuk menyimpan nama-nama hari dalam sepekan. Dari tabel tersebut dapat dilihat bahwa tipe data untuk field *Hari* adalah TINYINT dengan lebar 1 (satu) digit dengan nilai 0 untuk hari Minggu, 1 untuk hari Senin, 3 untuk hari Selasa dan seterusnya. Pemilihan tipe data TINYINT adalah untuk alasan efisiensi karena tipe data ini memiliki cakupan (*range*) dari 0 s.d. 255. Field *NamaHari* memiliki tipe data VARCHAR dengan lebar 6 (enam) digit karena field ini diisi dengan nama-nama hari yang panjang maksimumnya adalah 6 (enam) digit untuk hari "MINGGU".

**Tabel 3. Slot Waktu per Hari**

| No. | Field | Tipe | Lebar | Keterangan |
|---|---|---|---|---|
| 1 | KodeLes | TINYINT | 2 | Primary Key |
| 2 | Masuk | VARCHAR | 1 | P=Pagi, M=Malam |
| 3 | JamMasuk | TIME | | |
| 4 | JamKeluar | TIME | | |

Tabel 3 ini berguna untuk menyimpan data slot waktu per hari. Dari tabel tersebut dapat dilihat bahwa tipe data untuk field *KodeLes* adalah TINYINT dengan lebar 2 (dua) digit dengan nilai dari 1 sampai dengan 19 sesuai dengan banyak slot waktu per hari. Field *Masuk* memiliki tipe data VARCHAR dengan lebar 1 (satu) digit karena nilainya akan diisi dengan "P" untuk slot masuk pagi dan "M" untuk slot masuk malam. Field *JamMasuk* dan *JamKeluar* akan diisi dengan nilai waktu awal dan akhir yang memiliki selisih 45 (empat puluh lima) menit untuk masing-masing slot waktu.

**Tabel 4. Slot Waktu per Pekan**

| No. | Field | Tipe | Lebar | Keterangan |
|---|---|---|---|---|
| 1 | KodeHariLes | TINYINT | 3 | AUTO_INCREMENT |
| 2 | Hari | TINYINT | 1 | Foreign Key untuk tabel Hari |
| 3 | KodeLes | TINYINT | 2 | Foreign Key untuk tabel Les |

Tabel 4 ini berfungsi untuk menyimpan slot waktu dalam sepekan yang merupakan gabungan slot waktu per hari sesuai dengan jumlah hari belajar yang ada yaitu dari hari Senin sampai dengan Sabtu. Dari tabel tersebut dapat juga dilihat bahwa field *KodeHariLes* memiliki tipe data TINYINT dengan lebar 3 (tiga) digit karena datanya akan diisi dengan nilai 1 sampai 109 sesuai dengan banyaknya slot dalam satu pekan. Field *Hari* akan berisi nilai 1 sampai dengan 6 sesuai dengan banyaknya hari belajar dalam sepekan yang merupakan kunci tamu bagi tabel Hari. Field *KodeLes* akan berisi nilai 1 sampai dengan 19 sesuai dengan banyaknya slot per hari yang merupakan kunci tamu bagi tabel Les.

**Tabel 5. Slot Waktu Kesediaan Dosen**

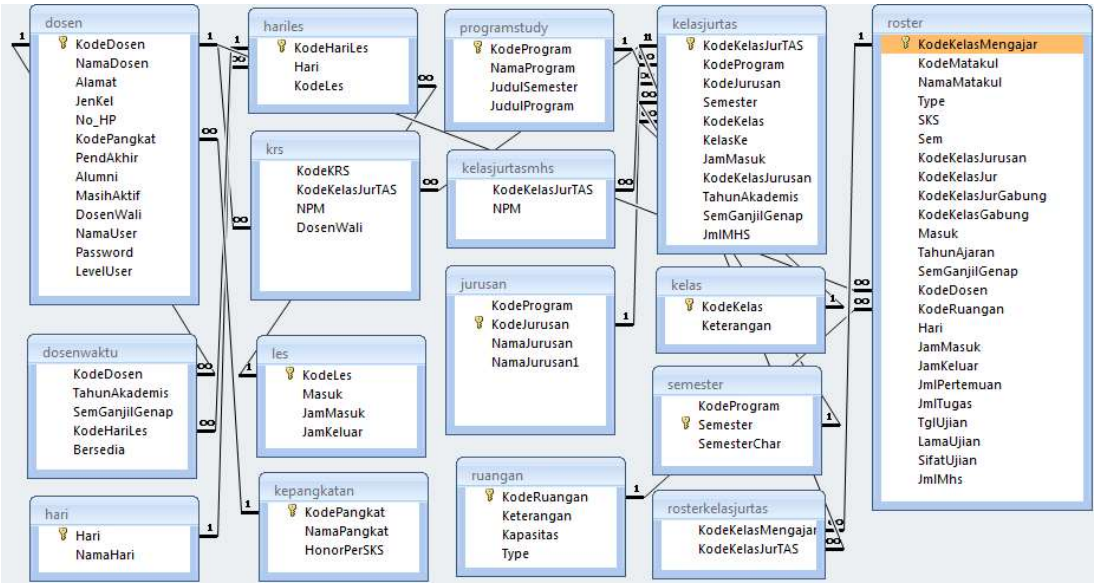
| No. | Field | Tipe | Lebar | Keterangan |
|---|---|---|---|---|
| 1 | KodeDosen | VARCHAR | 5 | Foreign Key untuk tabel Dosen |
| 2 | TahunAkademis | YEAR | 4 | |
| 3 | SemGanjilGenap | TINYINT | 1 | |
| 4 | KodeHariLes | TINYINT | 3 | Foreign Key untuk tabel HariLes |

| | | | | |
|---|---|---|---|---|
| 5 | Bersedia | TINYINT | 1 | |

Tabel 5 ini berguna untuk menyimpan kesediaan dosen dalam waktu sepekan yang merupakan gabungan tabel slot waktu sepekan dengan data dosen disertai tahun akademis dan keterangan semester. Dari tabel tersebut dapat dilihat bahwa field *KodeDosen* memiliki tipe data VARCHAR dengan lebar 5 (lima) sesuai dengan tipe data dan lebar yang ada pada tabel Dosen karena merupakan kunci tamu bagi tabel Dosen tersebut. Field *TahunAkademis* memiliki tipe data YEAR dengan lebar 4 (empat) digit akan diisi dengan nilai Tahun Akademis yang sedang berjalan. Field *SemGanjilGenap* memiliki tipe data TINYINT dengan lebar 1 (satu) digit akan diisi dengan nilai 1 untuk semester Ganjil dan 2 untuk semester Genap. Field *KodeHariLes* memiliki tipe data TINYINT dengan lebar 3 (tiga) digit akan diisi dengan nilai 1 sampai 109 sesuai dengan banyaknya slot dalam satu pekan dan merupakan kunci tamu bagi tabel HariLes.

**b. *Entity Relationship Diagram***

Seiring dengan penambahan 4 (empat) buah tabel database pada sistem yang sedang berjalan, menyebabkan terjadinya perubahan terhadap relasi tabel-tabel yang ada dalam database. Relasi antar tabel yang ada dalam database dapat dilihat pada gambar 7.




**Gambar 4. *Entity Relationship Diagram (ERD)* Sistem Penjadwalan Perkuliahan pada AMIK INTeL Com GLOBAL INDO**

Dari gambar 4 tersebut dapat dilihat bahwa terdapat 4 (empat) buah tabel tambahan sebagaimana yang diusulkan oleh peneliti yaitu tabel *Hari*, *Les*, *HariLes* dan *DosenWaktu.*

Tabel *Hari* memiliki relasi *one-to-many* terhadap tabel *HariLes*, di mana field *Hari* pada tabel *Hari* merupakan *primary key* sedangkan field *Hari* pada tabel *HariLes* merupakan *foreign key.* Tabel *Les* juga memiliki relasi *one-to-many* terhadap tabel *HariLes*, di mana field *KodeLes* pada tabel *Les* merupakan *primary key* sedangkan field *KodeLes* pada tabel *HariLes* merupakan *foreign key.* Ketiga tabel ini yaitu Tabel *Hari, Les* dan *HariLes* bertujuan untuk membedakan slot waktu (les) antara satu hari dengan hari lainnya.

Tabel *HariLes* memiliki relasi *one-to-many* terhadap tabel *DosenWaktu*, di mana field *KodeHariLes* pada tabel *HariLes* merupakan *primary key* sedangkan field *KodeHariLes* pada tabel *DosenWaktu* merupakan *foreign key*. Kedua tabel ini berfungsi untuk membedakan kesediaan waktu antara satu dosen dengan dosen lainnya.

**c. Disain Input**

Disain input untuk data hari dalam sepekan, data slot waktu untuk satu hari, data slot waktu untuk satu pekan, data slot waktu kesediaan dosen akan diuraikan sebagai berikut:

**1) Data Hari dalam Sepekan**

Karena hari dalam sepekan praktis tidak akan berubah sepanjang masa maka peneliti menganggap tidak perlu membuat disain input untuk data hari. Setelah tabel dibuat, maka untuk mengisi tabel hari cukup dengan membuat *query* sebagai berikut:

```
INSERT INTO Hari VALUES (0,'Minggu'), (1,'Senin'), (2,'Selasa'),
(3,'Rabu'), (4,'Kamis'), (5,'Jum`at'), (6,'Sabtu');
```

**2) Data Slot Waktu per Hari**

Sebagaimana yang telah dijelaskan pada aturan penjadualan bahwa data slot waktu per hari adalah 19 slot didapat dari 13 slot untuk kelas pagi dan 6 slot untuk kelas malam, maka disain input untuk data slot waktu per hari dapat dilihat pada gambar 5.

| Kode Les | Pagi/Malam | Jam Masuk | Jam Keluar | Aksi |
|---|---|---|---|---|
| 99 | xxxxx | 99:99 | 99:99 | ✔ |
| 99 | xxxxx | 99:99 | 99:99 | ✔ |
| 99 | xxxxx | 99:99 | 99:99 | ✔ |
| … | … | … | … | … |
| 99 | xxxxx | 99:99 | 99:99 | ✔ ✘ |
| | xxxxx | 99:99 | 99:99 | Insert |

**Gambar 5. Disain Input Data Slot Waktu Per Hari**

Dari gambar 5. bisa dilihat bahwa input data untuk kode les tidak ada (tidak perlu). Karena field *KodeLes* pada disain tabelnya diset AUTO_INCREMENT, maka nilainya akan berubah otomatis dari 1, 2, 3 dan seterusnya. Untuk kolom *Masuk* pilihannya adalah berupa *combo box* di mana terdapat 2 (dua) pilihan yaitu Pagi dan Malam. Untuk kolom *Jam Masuk* dan *Jam Keluar* diisi dengan nilai waktu dengan format "HH:mm" (huruf besar HH menyatakan format 24 jam). Pada kolom *Aksi* terdapat 2 (dua) gambar yaitu gambar ceklist dan gambar silang serta 1 (satu) buah tombol insert. Gambar ceklist adalah pilihan untuk menyimpan data sedangkan gambar silang adalah untuk menghapus data. Agar urutan waktunya tidak kacau jika dihapus, maka penghapusan hanya boleh dilakukan pada urutan terakhir saja. Tombol insert berguna untuk menambah data slot waktu per hari dengan terlebih dahulu memilih kolom *Masuk*, *Jam Masuk* dan *Jam Keluar*.

**3) Data Slot Waktu per Pekan**

Data slot waktu per pekan adalah merupakan *query* gabungan antara tabel hari dalam sepekan (Hari) dan table slot waktu per hari (Les). Agar isi tabel ini sinkron dengan kedua tabel tersebut, maka pembuatannya harus menggunakan *query* sebagai berikut :

```
INSERT INTO Hari HariLes select 0 as KodeHariLes, Hari, KodeLes from
Hari, Les where hari>0 order by Hari, KodeLes;
```

Dari *query* tersebut bisa dilihat bahwa data slot waktu per pekan ini mengacu kepada data slot waktu per hari. Oleh karena itu jika terjadi perubahan data slot waktu per hari, maka sistem penjadualan perkuliahan yang akan dirancang akan melakukan perubahan secara terprogram terhadap isi dari tabel slot waktu per pekan sesuai dengan data slot waktu per hari.

**4) Data Slot Waktu Kesediaan Dosen**

Data slot waktu kesediaan dosen adalah juga merupakan *query* gabungan antara tabel Dosen dan tabel slot waktu per pekan (HariLes). Agar isi tabel ini sinkron dengan kedua tabel tersebut, maka pembuatannya pun harus menggunakan *query* sebagai berikut:

```
Insert into DosenWaktu  Select Distinct a.KodeDosen,
c.TahunAkademis, c.SemGanjilGenap, b.KodeHariLes, 0 as Bersedia
from Dosen a, HariLes b, Setup c Where a.MasihAktif=1
Order By a.KodeDosen, b.KodeHariLes;
```

Dari *query* tersebut bisa dilihat bahwa data slot waktu kesediaan dosen ini juga mengacu kepada data slot waktu per hari. Oleh karena itu jika terjadi perubahan data slot waktu per hari, maka

sistem penjadualan perkuliahan yang akan dirancang pun akan melakukan perubahan secara terprogram terhadap isi dari tabel slot waktu kesediaan dosen sesuai dengan data slot waktu per hari.

Rancangan input untuk data slot waktu kesediaan dosen ini nantinya akan digabungkan dengan form input data dosen. Supaya tampilannya tidak terlalu panjang ke bawah karena memiliki 109 slot waktu, maka dalam penginputannya dibagi per hari. Untuk itu dibuat 6 (enam) buah link hari dari Senin sampai dengan Sabtu yang akan menampilkan data slot waktu kesediaan dosen per hari. Untuk lebih jelasnya dapat dilihat pada gambar 10 berikut.

***Senin*** Selasa Rabu Kamis Jum'at Sabtu

| Kode Hari Les | Hari | Jam Masuk | Jam Keluar | Bersedia |
|---|---|---|---|---|
| 99 | XXXXXX | 99:99 | 99:99 | ☑ |
| 99 | XXXXXX | 99:99 | 99:99 | ☑ |
| 99 | XXXXXX | 99:99 | 99:99 | ☑ |
| … | … | … | … | … |
| 99 | XXXXXX | 99:99 | 99:99 | ☑ |

**Gambar 6. Disain Input Data Slot Waktu Kesediaan Dosen**

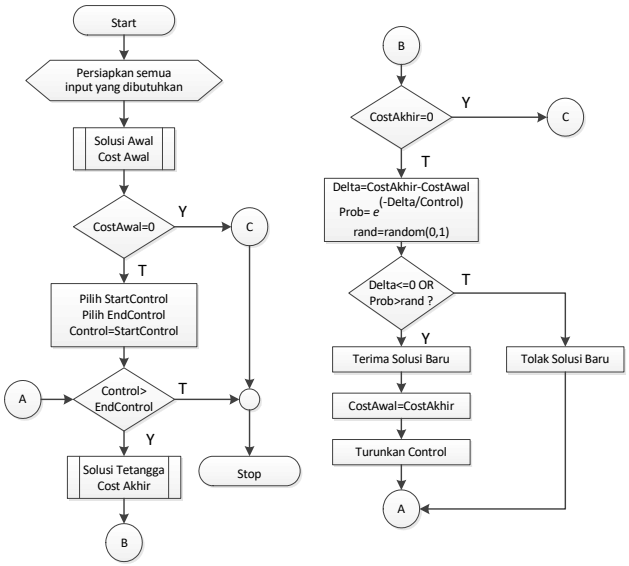
Dari gambar 6 di atas dapat dilihat bahwa kesediaan dosen pada slot waktu tertentu diimplementasikan dengan sebuah pilihan *check box*. Jika dosen bersedia mengajar pada slot waktu yang telah ditentukan maka pilihan *check box* harus dalam keadaan dicentang/dicontreng sebaliknya jika tidak bersedia *check box* harus dalam keadaan tidak dicentang/dicontreng.

**d. Disain Output**

Karena yang menjadi target dalam optimasi penjadualan perkuliahan dengan metode *Simulated Annealing* adalah jadual perkuliahan yang optimal maka peneliti menganggap bahwa tidak ada perubahan terhadap output yang dihasilkan oleh sistem yang sedang berjalan. Dengan kata lain disain output adalah tetap sebagaimana yang ditunjukkan oleh Tabel 1.

**e. Disain Program *Simulated Annealing***

Flow Chart Fungsi *Simulated Annealing* dapat dilihat pada gambar 7 berikut:




**Gambar 7. Flow Chart Fungsi *Simulated Annealing***

## 4. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan peneliti dalam membuat aplikasi optimasi penjadualan perkuliahan dengan metode *Simulated Annealing*, terdapat kelebihan dan kekurangan. Kelebihan dari aplikasi yang sudah dibuat adalah kemampuannya untuk menghasilkan beberapa kandidat jadual dalam waktu yang tidak terlalu lama untuk memberikan pilihan solusi penjadualan yang optimal. Di samping itu aplikasi menyediakan beberapa alat bantu (*tools*) pengujian untuk mengetahui seberapa optimal penjadualan perkuliahan yang dihasilkan berdasarkan batasan masalah

dan *constraint* yang telah ditentukan. Sedangkan kekurangan dari aplikasi ini adalah *tools* yang ada belum bisa melakukan pengujian terhadap semua *constraint* yang ada. Pengujian baru bisa dilakukan terhadap *Hard Constraint* sedangkan pengujian terhadap *Soft Constraint* belum ada.

Di samping kelebihan dan kekurangan yang telah diuraikan di atas, peneliti juga akan menguraikan kesimpulan yang dapat ditarik dari penelitian ini dan akan memberikan sumbangan pemikiran berupa saran-saran yang dianggap bermanfaat bagi dunia pendidikan pada umumnya dan khususnya bagi siapa saja yang akan melanjutkan penelitian ini di masa yang akan datang.

Berdasarkan hasil penelitian yang dilakukan oleh peneliti dalam melakukan proses optimasi penjadualan perkuliahan dengan metode *Simulated Annealing,* maka dapat ditarik beberapa kesimpulan sebagai berikut:

a. Metode *Simulated Annealing* adalah suatu metode pencarian heuristik yang memiliki suatu kelebihan dalam pembangkitan solusi tetangga (*neighborhood*) dalam proses penyelesaian iterasinya. Kelebihan metode *Simulated Annealing* adalah memberikan peluang yang sebesar-besarnya kepada peneliti yang ingin menyelesaikan studi kasus dengan metode *Simulated Annealing* untuk mendefinisikan fungsi solusi tetangga sendiri.
b. Pemrograman Java yang bersifat *Object Oriented Programming* (OOP) sangat cocok digunakan untuk menyelesaikan masalah-masalah optimasi terutama yang berhubungan dengan manipulasi atau pengolahan database dengan tidak melibatkan *Graphical User Interface* (GUI).

## DAFTAR PUSTAKA

[1] Srinivas, P.S.; Raju V. Ramachandra; Rao C.S.P., *Particle Swarm Optimization Approach for Scheduling of Flexible Job Shops,* International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 1, Issue 5, 2012.

[2] Safwan M. Shatnawi, Fawzi Albalooshi, and Khaleel Rababa'h, *Generating Timetable and Students Schedule Based on Data Mining Techniques,* International Journal of Engineering Research and Applications (IJERA), Vol. 2, Issue 4, pp. 1638-1644, 2012.

[3] Filik Ümmühan Başaran and Kurban Mehmet, *Solving Unit Commitment Problem Using Modified Subgradient Method Combined with Simulated Annealing Algorithm,* Hindawi Publishing Corporation Mathematical Problems in Engineering, Article ID 295645, 2010.

[4] Vishwakarma Kamlesh Kumar, Dubey Hari Mohan, Pandit Manjere and Panigrahi B.K., *Simulated Annealing Approach for Solving Economic Load Dispatch Problems with Valve Point Loading Effects,* International Journal of Engineering, Science and Technology, Vol. 4, No. 4, pp. 60-72, 2012.

[5] Nasiri M., Taghavi Sadat L., Minaee, B., *Numeric Multi-Objective Rule Mining Using Simulated Annealing Algorithm,* International Journal of Applied Operational Research Vol. 1, No. 2, pp. 37-48, 2010.

[6] Qin Jin, Ni Ling-lin, dan Shi Feng, *Combined Simulated Annealing Algorithm for the DiscreteFacility Location Problem,* The ScientificWorld Journal Volume 2012, Article ID 576392, 2012.

[7] Burke Edmund, Bykov Yuri, Newall James, Petrovic Sanja, *A Time-Predefined Approach to Course Timetabling,* Yugoslav Journal of Operations Research,Vol. 13, Numb. 2, pp. 139-151, 2003.

[8] Defu, Zhang; Yongkai, Liu; Rym M'Hallah; and Stephen, Leung, C.H., *A Simulated Annealing with a New Neighbourhood Structure Based Algorithm for High School Timetabling Problems*, European Journal of Operational Research, Vol. 203, pp. 550–558, 2010.

[9] Jhonson, David, S., *A Brief History of NP-Completeness, 1954–2012*, Documenta Mathematica, Extra Volume ISMP, pp. 359–376, 2012.