OPTIMASI PENJADWALAN MATA KULIAH DI JURUSAN TEKNIK INFORMATIKA PENS DENGAN MENGGUNAKAN ALGORITMA PARTICLE SWARM OPTIMIZATION (PSO)

Dian Ariani[1], Arna Fahriza,S.Kom,M.Kom[2], Ira Prasetyaningrum,S.Si,M.T[3]
Mahasiswa Jurusan Teknik Informatika[1] , Dosen Pembimbing [2], Dosen Pembimbing [3]
Politeknik Elektronika Negeri Surabaya
Institut Teknologi Sepuluh Nopember
Kampus PENS-ITS Keputih Sukolilo Surabaya 60111
Telp (+62)31-5947280, 5946114, Fax. (+62)31-5946114
Email : ariani_ss@yahoo.com

ABSTRAK

Di perguruan tinggi, program penjadwalan merupakan salah satu hal penting dalam proses belajar mengajar, karena semua kegiatan dosen dan mahasiswa bergantung pada jadwal yang ada, sehingga harus disusun dengan benar dan diperbaiki pada awal tahun akademik, sehingga nantinya tidak mengganggu aktifitas belajar mengajar antar dosen dan mahasiswa.

Untuk menyelesaikan masalah tersebut dalam tugas akhir ini digunakan algoritma Particle Swarm Optimized (PSO) utnuk melakukan optimasi pada jadwal kuliah. Karena algoritma PSO memiliki tool-tool yang cukup handal dengan penggunaan yang cukup mudah.

Tugas Akhir ini ini bertujuan untuk membuat suatu sistem komputasi untuk menggantikan penjadwalan secara manual, dan hasil akhirnya diharapkan dapat mengatur jam mengajar dosen dan juga jadwal perkuliahan mahasiswa, sehingga menghindari adanya bentrokan jadwal

Kata kunci : *Particle Swarm Optimizaton,* PSO, penjadwalan

1. PENDAHULUAN

1.1 Latar Belakang

Di perguruan tinggi, program penjadwalan merupakan salah satu hal penting dalam proses belajar mengajar, karena semua kegiatan dosen dan mahasiswa bergantung pada jadwal yang ada, sehingga harus disusun dengan benar dan diperbaiki pada awal tahun akademik, sehingga nantinya tidak mengganggu aktifitas belajar mengajar antar dosen dan mahasiswa.

Selama ini penjadwalan mata kuliah di Politeknik Elektronika Negeri Surabaya (PENS) masih dilakukan secara manual, sedangkan untuk membagi dosen sesuai dengan bidang mata kuliahnya dan waktu tertentu diperlukan pengaturan yang cukup rumit.

Oleh karena itu dalam tugas akhir ini dibuat suatu sistem komputasi yang dapat menggantikan cara manual tersebut dan dapat menghasilkan hasil yang lebih bagus dan waktu yang lebih singkat, hanya dengan memberikan parameter input yang dibutuhkan, dapat menghasilkan output penjadwalan mata kuliah yang diinginkan.

Untuk menyelesaikan masalah tersebut digunakan algoritma Particle Swarm Optimized (PSO) karena algoritma PSO memiliki tool-tool yang cukup handal dengan penggunaan yang cukup mudah.

Tugas akhir ini nantinya diharapkan dapat mengatur jam mengajar dosen dan juga jadwal perkuliahan mahasiswa, sehingga menghindari adanya bentrokan jadwal.

1.2 PERUMUSAN MASALAH

Berdasarkan uraian diatas, maka permasalahan yang timbul dalam pengerjaan tugas akhir ini adalah bagaimana membuat jadwal mata kuliah dan pembagian dosen pada tiap-tiap kelas di PENS, sehingga didapatkan kombinasi mata kuliah yang lebih baik guna menghasilkan jadwal mata kuliah yang optimal. Jadwal mata kuliah dikatakan optimal apabila tidak didapatkan kress didalamnya dan tidak melanggar konstrain yang ditentukan.

1.3 BATASAN MASALAH

Adapun batasan masalah dalam tugas akhir ini adalah :

1. Mata kuliah yang akan dijadwalkan dalam tugas akhir ini hanya pada jurusan Teknik Informatika PENS.

2. Atribut yang digunakan dalam pembuatan jadwal mata kuliah ini meliputi dosen, mahasiswa, dan mata kuliah.

Algoritma yang digunakan untuk menyelesaikan permasalahan tugas akhir penjadwalan ini adalah Particle Swarm Optimization

1.4 TUJUAN DAN SASARAN

Tujuan dari tugas akhir yang diusulkan ini adalah:

1. Menghasilkan suatu aplikasi yang berfungsi untuk menyusun jadwal mata kuliah di jurusan Teknik Informatika PENS.
2. Mengembangkan proses komputasi dengan Particle Swarm Optimiztion untuk menyelesaikan permasalahan penjadwalan mata kuliah di jurusan Teknik Informatika PENS.
3. Mempermudah dan mempercepat proses pembuatan jadwal kuliah di jurusan Teknik Informatika PENS dengan hasil yang optimal.

2. DASAR TEORI

2.1 PARTICLE SWARM OPTIMIZTION (PSO)

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

Algoritma dasar PSO terdiri dari tiga tahap, yaitu pembangkitan posisi serta kecepatan partikel, *update velocity* (*update* kecepatan), *update position* (*update* posisi). Partikel berubah posisinya dari suatu perpindahan (iterasi) ke posisi lainnya berdasarkan pada *update velocity*. 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 (1.1) dan (1.2).

$$x_0^i = x_{\min} + rand(x_{\max} - x_{\min}) \qquad (1.1)$$

$$v_0^i = x_{\min} + rand(x_{\max} - x_{\min}) \qquad (1.2)$$

Posisi dan kecepatan direpresentasikan dalam bentuk vektor dimana *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 :

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

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

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 selnjutnya. Perumusan *update velocity* mencakup beberapa parameter random, rnd, untuk mendapatkan cakupan yang baik pada *design space*, tiga parameter yang mempengaruhi arah pencarian, yaitu *inertia factor* (w), *self confidence* (c1), *swarm confidence* (c2) akan digabungkan dalam satu penyajian, seperti yang ditunjukkan persamaan berikut :

$$v_{k+1}^i = w * v_k^i + c1 * rnd * \left(p^i - x_k^i\right) + c2 * rnd * \left(p_k^g - x_k^i\right) \qquad (1.3)$$

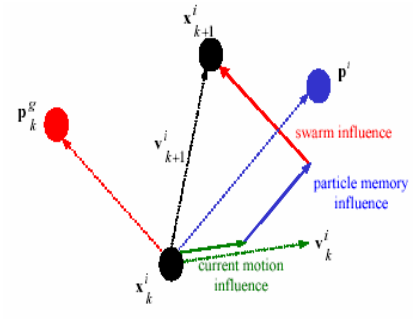
dengan range w = 0.4 – 1.4, $c1$ = 1.5 – 2.0 dan $c2$ = 2.0 – 2.5

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

$$x_{k+1}^i = x_k^i + v_{k+1}^i \qquad (1.4)$$

Proses update posisi dan kecepatan secara jelas digambarkan pada Gambar 1.1

Tiga tahapan diatas akan diulang sampai kriteria 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.



Gambar 1.1 update posisi dan kecepatan PSO

Pseudo Code Algoritma PSO

```
for setiap partikel
   Inisialisasi partikel menggunakan persamaan (1.1) dan (1.2)
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
   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 (1.3)
        Update posisi partikel menggunakan persamaan (1.4)
     end
  until (KriteriaBerhenti == true)
```

2.2 PENJADWALAN

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 mata kuliah, penjadwalan ujian, atau bisa juga penjadwalan karyawan, baik dalam suatu perusahaan ataupun dalam rumah sakit. Dalam penjadwalan kuliah, akan dibahas tentang pembagian jadwal untuk tiap mahasiswa pada kuliah tertentu sekaligus dosen pengajarnya, dalam penjadwalan pelajaran sekolah akan dibahas tentang pembagian jadwal pelajaran untuk tiap-tiap kelas yang ada beserta guru pengajar pelajaran tersebut, dalam penjadwalan ujian akan dibahas pengaturan dosen yang menjaga ujian dan mahasiswa atau murid yang menempati ruang ujian yang ada, sedangkan pada penjadwalan karyawan, dilakukan pengaturan karyawan yang akan bekerja pada waktu tertentu di bagian tertentu.

Proses tersebut tentu saja dibuat berdasarkan permasalahan yang ada. Beberapa proses umum ynag perlu dilakukan untuk menyelesaikan suatu proses penjadwalan menurut Research Group – Computer Science (BGU) adalah:

1. 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.
2. Mendesign metode penyelesaian untuk permasalahan penjadwalan tersebut.
   Dari model yang telah ada, ditentukan metode yang akan digunakan untuk menyelesaikan permasalahan penjadwalan tersebut.
3. 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.

Sedangkan pembahasan penjadwalan menurut Tomas Muller dan Roman Bartak sebagai berikut :

1. Aktivitas yang dilakukan
   Maksudnya adalah bahwa penentuan dari permasalahan penjadwalan yang dibahas. Misalnya penjadwalan mata kuliah di perguruan tinggi.
2. Sumber-sumber yang dipakai
   Sumber-sumber yang dipakai berarti hal-hal yang dapat digunakan untuk menyelesaikan permasalahan penjadwalan (aktifitas) yang telah ditentukan atau bisa juga dikatakan sebagai data yang digunakan. Misalnya pada penjadwalan mata kuliah diperlukan data mata kuliah, dosen, kelas dan sumber lain yang diperlukan.
3. Syarat-syarat yang diperlukan
   Syarat disini adalah hal-hal yang perlu diperhatikan ketika menyusun suatu penjadwalan. Misalnya saja dalam penjadwalan mata kuliah terdapat syarat bahwa seorang dosen tidak boleh mengajar dua kelas yang berbeda dalam waktu / jam kuliah yang sama.
4. Hubungan Timbal Balik
   Yang dimaksud hubungan timbal balik disini adalah bagaimana jadwal yang telah dibuat tersebut dapat sesuai dengan yang diinginkan oleh user.

2 .3 SEKILAS TENTANG JAVA

Java.lang dan java.util merupakan dua package yang terdapat di dalam java. Package java.lang memuat sejumlah class penting, termasuk class-class wrapper dan package ini bisa langsung

diimport secara otomatis tanpa mendeklarasikannya. Sedangkan yang tidak kalah pentingnya adalah package java.util. Package ini juga memuat ArrayList

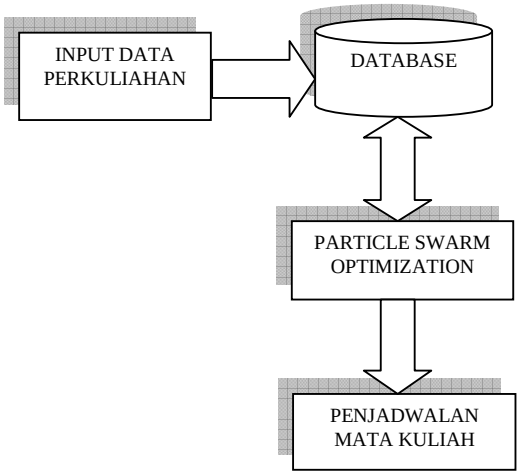
2.3.1 ArrayList

Kelas *ArrayList* merupakan implementasi dari *interface List*. Kelas ini mendukung array yang dinamis dan dapat digunakan sesuai dengan kebutuhan. Dalam java, standar panjang sebuah array tetap. Setelah array diciptakan, mereka tidak bisa bertambah atau berkurang, yang berarti Anda harus mengetahui terlebih dahulu berapa banyak elemen-elemen array akan Anda dibuat. Namun, kadang-kadang Anda mungkin tidak tahu seberapa besar array yang Anda butuhkan. Untuk mengatasi situasi ini, diciptakan *ArrayList*. dimana pada *ArrayList* tersebut terdapat objek yang dinamis dan dapat menambah atau mengurangi ukuran elemen-elemen di dalamnya sesuai dengan kebutuhan yang diperlukan.

- Objek ArrayList selalu memiliki ukuran tertentu, dan tidak boleh mengambil posisi di luar ukuran ArrayList. Dalam hal ini, ArrayList mirip seperti array biasa. Akan tetapi, ukuran ArrayList bisa bertambah kapan saja jika diperlukan.

3. PERANCANGAN SISTEM

3.1 PERANCANGAN SISTEM



Gambar 3.1 Konfigurasi Sistem

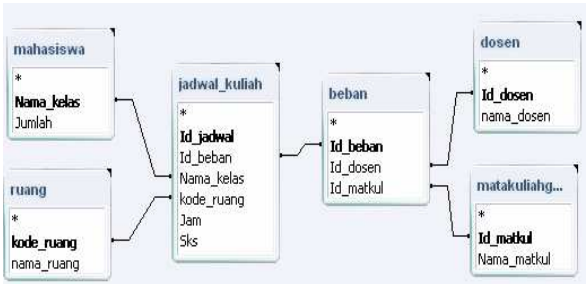
Sistem penjadwalan ini nantinya hanya dapat digunakan oleh petugas BAAK, yang bertugas dalam penyusunan sebuah jadwal mata kuliah di PENS khususnya jurusan Teknik Informatika, jadwal dibuat dengan ketentuan :

1. Tidak ada bentrok mahasiswa, yaitu setiap mahasiswa hanya dapat mengikuti satu mata kuliah dalam hari dan jam yang sama.
2. Tidak ada bentrok dosen, yaitu setiap dosen hanya bisa mengajar di satu kelas pada hari dan jam yang sama.
3. Setiap dosen hanya bisa mengajar pada suatu kelas sebanyak satu kali pada hari yang sama.
4. Tidak boleh ada bentrok penggunaan lab.

3.1.1 Database

3.1.1.1 Entity Relationship Diagram(ERD)

Entity Relationship diagram dari system penjadawalan yang dikerjakan memiliki 3 entitas dan digambarkan pada gambar 3.2 :



Gambar 3.2 ER Diagram

Dari ER Diagram yang telah digambarkan diatas , didapatkan 4 tabel, yaitu 5 tabel murni dan satu table yang menghubungkan antara table dosen, matakuliah dan mahasiswa.

3.1.1.2 Perancangan Tabel

Untuk rincian isi dari table- table yang akan digunakan adalah sebagai berikut :

1. Tabel Matakuliah

Tabel 3.1 Tabel Matakuliah

Nama Field	Tipe Data
Id	Number Primary Key
Nama_Matkul	Text(50)

2. Tabel Mahasiswa

Tabel 3.2 Tabel Mahasiswa

Nama Field	Tipe Data
Nama_kelas	Text(10)
Jumlah	Number

3. Tabel Ruang

Tabel 3.3 Tabel Ruang

Nama Field	Tipe Data
Kode_Ruang	Text(5)
Nama_Ruang	Text(50)

4. Tabel Dosen

Tabel 3.4 Tabel Dosen

Nama Field	Tipe Data
Id_Dosen	Number
Nama_Dosen	Text(50)

5. Tabel Beban

Tabel 3.5 Tabel Beban

Nama Field	Tipe Data
Id_Beban	Number
Id_Dosen	Number
Id_matkul	Number

6. Tabel Jadwal

Tabel 3.6 Tabel Jadwal_Kuliah

Nama Field	Tipe Data
Id_jadwal	Number
Mata_kuliah	Number
Mahasisawa	Number
Jam	Number
SKS	Number

3.1.1.3 Koneksi Database

Dalam tugas akhir ini database yang digunakan adalah MySql, untuk melakukan koneksi ke java 3 langkah dalam menyetting datasource yaitu

- mendaftarkan file JAR yang berisi JDBC driver dengan container.
- Membuat connection pool ke database
- Mendaftarkan sebuah datasource yang digunakan untuk connection pool.

Sedangkan langkah-langkah untuk menambahkan source codenya adalah sebagai berikut :

1. Load Driver JDBC.
2. Definisikan URL Database.
3. Membuat dan Melakukan Koneksi.
4. Menutup Koneksi.

3.1.2 Graphic User Interface (GUI)

Sistem Aplikasi Penjadwalan Matakuliah ini hanya terdiri dari satu form utama untuk menjalankan seluruh proses penjadwalan dan menampilkannya dalam bentuk table-tabel yang berisi mata kuliah dan dosen pengajar yang disajikan perkelas.

3.1.2.1 Form Utama

Form utama merupakan inti dari aplikasi ini, Pada form utama ini terdapat 3 button utama.

Button utama pada form utama ini merupakan button yang menjalan system aplikasi ini :

1. Button Initialize
2. Button Iterate
3. Button Do it All

Selain button utama juga terdapat satu menu utama, yaitu menu konfigurasi, untuk merubah nilai parameter sesuai inputan user.

Berikut ini adalah menu utama yang digunakan untuk memprosen penjadwalan, hingga menghasilkan suatu jadwal yang optimal dan menampilkannya dalam table.

Gambar 3.4 FormUtama

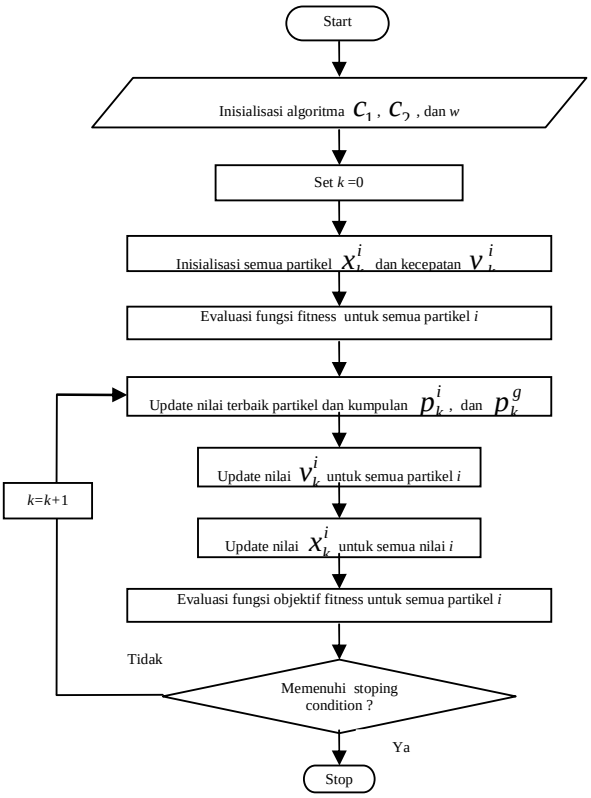
Keterangan :

1. Button Do it All, untuk menjalankan keseluruhan proses penjadwalan
2. Fitness yang dihasilkan.
3. Total iterasi dari keseluhan proses
4. Button Initialize, untuk membangkitkan partikel awal. Digunakan jika user ingin menjalankan aplikasi per step iterasi
5. Button Iterate, yang akan muncul setelah button Initialize ditekan, digunakan untuk menjalankan aplikasi per step iterasi
6. Kelas-kelas yang dijadwalkan
7. Tabel yang berisi hasil penjadwalan Matakuliah
8. Convert jadwal ke file pdf
9. Menu konfigurasi, untuk melakukan konfigurasi parameter.

Message Dialog untuk konfigurasi parameter berikut akan muncul setelah menu konfigurasi dipilih

3.1.3 Algoritma Particle Swarm Optimization(PSO)

Pada PSO akan digunakan algoritma dasar sederhana dari PSO dengan variabel bernilai integer dan faktor inersia w statis, tahapan-tahapan secara jelas seperti digambarkan pada diagram alir dibawah ini :



Gambar 3.5 Diagram alir PSO

3.2 PERANCANGAN DAN IMPLEMENTASI PROGRAM

3.2.1 Pembangkitan posisi dan velocity awal partikel

Dalam penjadwalan kuliah ini posisi dan velocity dimisalkan sebagai slot.

Posisi dan velocity awal ditentukan secara random dengan batasan nilai minimum slot dan maximum slot (0-14). Nilai posisi dan velocity awal partikel diinisialisasi sama, sehingga hanya melakukan satu kali 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 15 slot dalam satu minggu perkuliahan, dimana setiap harinya terdapat 3 slot yang dapat digunakan dan setiap slot mewakili 3 jam kuliah. gambaran dari slot yang akan digunakan untuk tiap partikel ditunjukkan pada table berikut :

Kelas 1 D3 TI A

Senin	Selasa	Rabu	Kamis	Jumat
0	3	6	9	12
1	4	7	10	13
2	5	8	11	14

•
•
•

Kelas 4 D3 TI B

Senin	Selasa	Rabu	Kamis	Jumat
0	3	6	9	12
1	4	7	10	13
2	5	8	11	14

Pembangkitan posisi dan velocity awal partikel dilakukan secara random dengan menggunakan persamaan (1.1) dan (1.2), dimana X_{min}(batas terkecil) adalah 0 dan X_{max}(batas tertinggi) adalah 14, kemudian mata kuliah dan dosen diletakkan pada posis slot yang didapatkan secara random.

3.2.2 Menentukan nilai fitness masing-masing partikel

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

Konstrain-konstrain 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 konstrain-konstrain diatas, maka nilai fitness masing-masing partikel akan diincrement sebanyak satu untuk tiap pelanggaran. Sehingga partikel terbaik adalah partikel dengan nilai fitness terkecil.

3.2.3 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
```

3.2.4 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 (1.3).

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
```

Sedangkan untuk proses update posisi digunakan persamaan (1.4). Cara update posisi yaitu dengan cara menukar posisi lama dengan posisi dibalakangnya sebanyak hasil update.

Pseudo code untuk update vector/posisi

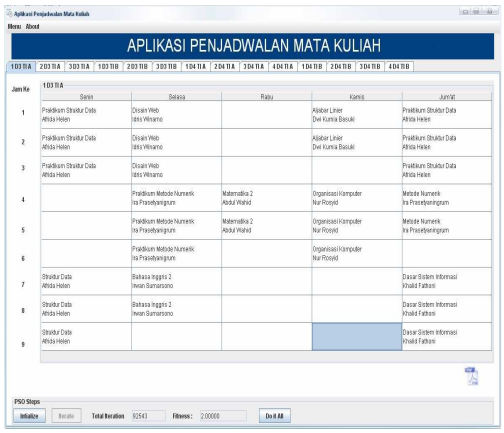
```
For j = 1 To size
   For i = 1 To vector
      particlevector(j, i) = particlevector(j, i)
+(particlevelocity(j, i)
   Next i
Next j
```

4. UJI COBA DAN ANALISA

4.1 UJI COBA

4.1.1 Hasil Running Aplikasi

Untuk menjalankan aplikasi penjadwalan ini user harus menekan button *Do it All* untuk menjalankan keseluruhan dari proses penjadwalan, mulai dari pembangkitan partikel jadwal awal hingga dihasilkan jadwal yang optimal, setelah itu aplikasi akan menampilkan hasil penjadwalan tersebut, fitness dan jumlah iterasi dari proses penjadwalan.



Gambar 4.1 Hasil Running Aplikasi Penjadwalan

4.1.2 Hasil Percobaan

- **Uji coba dengan parameter 1**

Parameter yang digunakan pada uji coba 1 ini ditunjukkan pada tabel berikut :

Tabel 4.1 Parameter 1

W	C1	C2	Iterasi Maksimum
0.9	1.5	1.5	100000

Uji coba 1 yang akan dilakukan pada pengujian aplikasi penjadwalan ini akan dilakukan pada 2 macam data inputan, yaitu

- Dengan menggunakan jumlah partikel sebanyak 5
- Dengan menggunakan jumlah partikel sebanyak 10

Setelah dilakukan beberapa kali percobaan pada 3 macam data inputan yang disebutkan diatas maka dihasilkan data-data pada tabel- tabel beikut ini :

Untuk percobaan dengan menggunakan 5 partikel dan dilakukan 10 kali percobaan didapatkan data sebagai berikut:

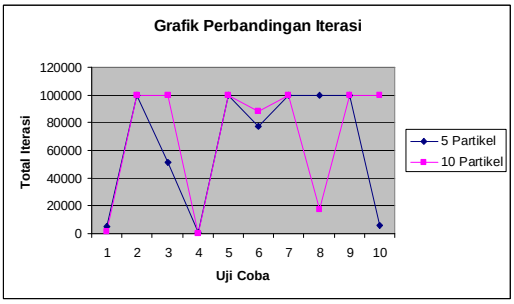
Tabel 4.2 Hasil Uji Coba parameter 1 menggunaan 5 Partikel

Uji Coba Ke-	Iterasi	Fitness
1	4978	0
2	100000	3
3	51004	0
4	413	0
5	100000	3
6	77535	0
7	100000	3
8	100000	3
9	100000	2
10	5478	0

Untuk percobaan dengan menggunakan 10 partikel dan dilakukan 10 kali percobaan didapatkan data sebagai berikut:

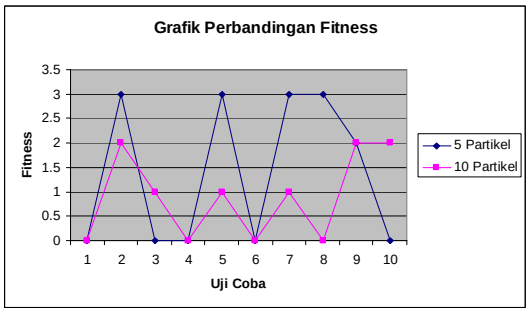
Tabel 4.3 Hasil Uji Coba parameter 1 dengan menggunaan 10 Partikel

Uji Coba Ke-	Iterasi	Fitness
1	1633	0
2	100000	2
3	100000	1
4	274	0
5	100000	1
6	88445	0
7	100000	1
8	17393	0
9	100000	2
10	100000	2



Gambar 4.2 Grafik Perbandingan Iterasi dengan Parameter 1

Hasil dari percobaan dengan menggunakan parameter 1 didapatkan jadwal dengan iterasi rata-rata hampir sama dengan menggunakan 5 partikel dan 10 partikel.



Gambar 4.3 Grafik Perbandingan Fitness dengan Parameter 1

Hasil dari percobaan dengan menggunakan parameter 1 didapatkan jadwal dengan nilai fitness rata-rata yang kecil dihasilkan oleh uji coba dengan menggunakan 10 partikel.

- **Uji coba dengan parameter 2**

Parameter yang digunakan dalam uji coba 2 ini ditunjukkan pada tabel berikut :

Tabel 4.4 Parameter 2

W	C1	C2	Iterasi Maksimum
0.5	1.5	1.5	100000

Uji coba 1 yang akan dilakukan pada pengujian aplikasi penjadwalan ini akan dilakukan pada 2 macam data inputan, yaitu

- Dengan menggunakan jumlah partikel sebanyak 5
- Dengan menggunakan jumlah partikel sebanyak 10

Setelah dilakukan beberapa kali percobaan pada 2 macam data inputan yang disebutkan diatas maka dihasilkan data-data pada tabel- tabel beikut ini :

Untuk percobaan dengan menggunakan 5 partikel dan dilakukan 10 kali percobaan didapatkan data sebagai berikut:

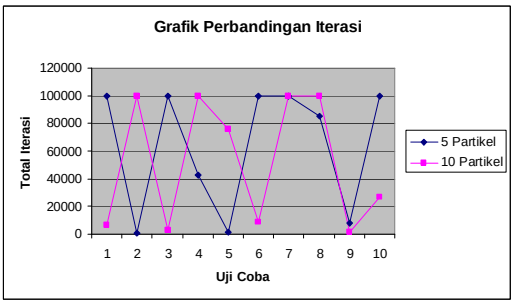
Tabel 4.5 Hasil Uji Coba parameter 2 dengan menggunaan 5 Partikel

Uji Coba Ke-	Iterasi	Fitness
1	100000	3
2	723	0
3	100000	2
4	42659	0
5	1258	0
6	100000	1
7	100000	1
8	85513	0
9	8266	0
10	100000	2

Untuk percobaan dengan menggunakan 10 partikel dan dilakukan 10 kali percobaan didapatkan data sebagai berikut:

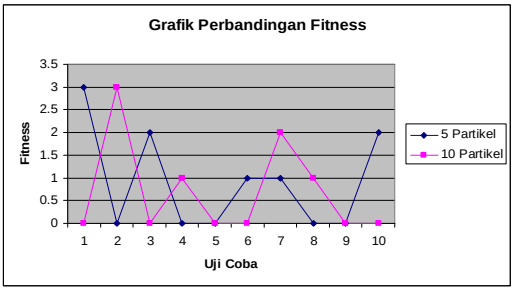
Tabel 4.6 Hasil UjiParameter 2 Coba dengan menggunaan 10 Partikel

Uji Coba Ke-	Iterasi	Fitness
1	6527	0
2	100000	3
3	3195	0
4	100000	1
5	76018	0
6	8496	0
7	100000	2
8	100000	1
9	1198	0
10	26956	0



Gambar 4.4 Grafik Perbandingan Itetasi dengan Parameter 2

Hasil dari percobaan dengan menggunakan parameter 2 didapatkan jadwal dengan iterasi rata-rata yang yang hampir sama dengan menggunakan 5 partikel dan 10 partikel.

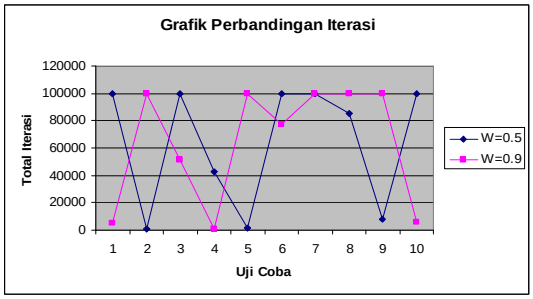


Gambar 4.5 Grafik Perbandingan Fitness dengan Parameter 2

Hasil dari percobaan dengan menggunakan parameter 2 didapatkan jadwal dengan nilai fitness rata-rata yang kecil dihasilkan oleh uji coba dengan menggunakan 10 partikel.

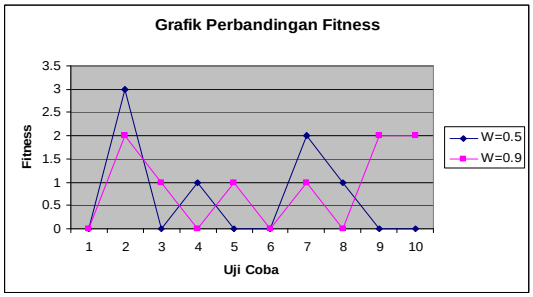
Grafik Perbandingan dengan nilai W yang berbeda

- Dengan menggunakan 5 partikel



Gambar 4.6 Grafik Perbandingan Iterasi 5 Partikel

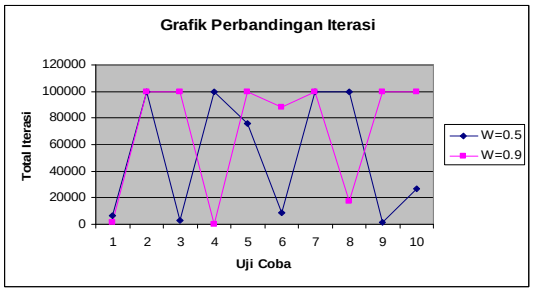
Hasil uji coba dengan parameter w = 0.5 menghasilkan iterasi rata-rata yang hampir sama dengan menggunakan w = 0.9 pada uji coba dengan 5 partikel.



Gambar 4.7 Grafik Perbandingan Fitness 5 Partikel

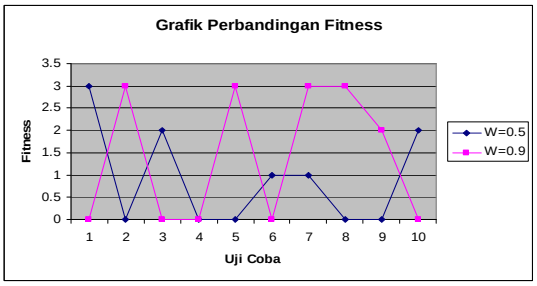
Hasil uji coba dengan parameter w = 0.5 menghasilkan fitness rata-rata lebih kecil daripada dengan menggunakan w = 0.9 pada uji coba dengan 5 partikel.

- Dengan menggunakan 10 partikel



Gambar 4.8 Grafik Perbandingan Iterasi 10 Partikel

Hasil uji coba dengan parameter w = 0.5 menghasilkan iterasi rata-rata yang hampir sama dengan menggunakan w = 0.9 pada uji coba dengan 10 partikel.



Gambar 4.9 Grafik Perbandingan Fitness 10 Partikel

Hasil uji coba dengan parameter w = 0.5 menghasilkan iterasi rata-rata lebih kecil daripada dengan menggunakan w = 0.9 pada uji coba dengan 10 partikel.

4.2 ANALISA HASIL PERCOBAAN

Dari uji coba yang telah dilakukan diatas dapat dilakukan analisa terhadap parameter-parameter yang digunakan. Analisa yang didapat sebagai berikut :

- Jumlah partikel yang dibangkitkan mempengaruhi optimasi jadwal yang dihasilkan, semakin banyak partikel yang terlibat maka dapat menghasilkan jadwal yang rata-rata hasilnya optimal, hasil ini dimungkinkan karena banyaknya partikel yang terlibat dalam proses penjadwalan dapat memberikan banyak pilihan partikel yang memiliki nilai fitness yang baik, yang kemudian akan diambil partikel dengan nilai fitness tarbaik.
- Sedangkan pada uji coba dengan nilai inertia factor (w)=0.5 dapat menghasilkan rata-rata jadwal yang lebih optimal dibanding dengan uji coba dengan menggunakan w=0.9, dengan nilai c1 dan c2 yang sama pada kedua percobaan tersebut.
- Parameter yang digunakan tidak mempengaruhi total iterasi, namun berpengaruh pada nilai fitness yang dihasilkan.

5. KESIMPILAN DAN SARAN

5.1 KESIMPULAN

Setelah dilakukan uji coba dan analisa terhadap tugas akhir ini, maka kita dapatkan :

1. Algoritma Particle Swarm Optimization (PSO) dapat digunakan untuk mengoptimasi permasalahan penjadwalan Matakuliah di jurusan Teknik Informatika PENS.
2. Hasil akhir dari penjadwalan matakuliah dengan PSO yang paling optimal yaitu dengan menggunaka parameter C1=1.5, C2= 1.5, W=0.5 dan jumlah partikel sebanyak 10.
3. Beberapa percobaan dapat menghasilkan nilai fitness minimum yang diharapkan, yakni 0 pada iterasi yang cukup besar, namum pada beberapa percobaan tidak dapat menghasilkan nilai fitness 0 pada iterasi maksimal yang ditentukan dikarenakan masih ada pelanggaran konstrain yang belum dapat dioptimasi.
4. Penjadwalan pada 14 kelas pada tugas akhir ini dapat 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 lab yang sama pada hari dan jam yang sama.

5.2 SARAN

Dengan melihat hasil yang didapatkan dari uji coba maka disarankan :

1. PSO dirasa cukup efisien untuk digunakan dalam masalah penjadwalan, namun parameter yang digunakan dapat diubah-ubah untuk mendapatkan jadwala yang lebih optimal
2. Penelitian mengenai performansi algoritma masih sangat dibutuhkan lebih lanjut pada bidang aplikasi lainnya, sehingga mampu memberikan kontribusi padsa perkembangan algoritma tersebut.
3. Konstrain dapat ditambah dan disesuaikan dengan kebutuhan dan persyaratan pembuatan jadwal yang berlaku.

6. DAFTAR PUSTAKA

1. Faradisa. Rosiyah, *Perbandingan Hasil Optimasi Particle Swarm Optimization (PSO) dan Genetic Algorithm (GA) pada Fungsi Rosenbrock (Banana Function)*, Tugas Akhir ITS, Surabaya, 2007.
2. Suyanto, Algotitma Optimasi Deterministik atau Probabilitik, Graha Ilmu, Yogyakarta, 2010.
3. http://java.lyracc.com/belajar/java-untuk-pemula/arraylists-dan-vector
4. Chu. Shu-Chuan, Chen. Yi-Tin and Ho. Jiun-Huei, *Timetable Scheduling Using Particle Swarm Optimization*, Cheng-Shiu University, Kaohsiung County 833, Taiwan, 2006

5. Agustina. Ira Lia, *Penjadwalan Pelajaran SMU Negeri Mojoagung dengan Algoritma,* Tugas Akhir ITS, Surabaya, 2005.
6. S.G. Ponnambalam, N. Jawahar, S. Chandrasekaran, *Discrete Particle Swarm Optimization Algorithm for Flowshop Scheduling*, Monash University, Thiagarajar College of Engineering, S R M V Polytechnic College, Malaysia, India