BAB 4 PERANCANGAN SISTEM

4.1 Formulasi Permasalahan

Pada bagian ini dibahas perancangan algoritme untuk mendapatkan solusi yang optimal. Tahap awal pada bagian ini yaitu melakukan pemetaan jadwal mahasiswa dan dosen. Tahap selanjutnya adalah memproses data ke dalam metode algoritme genetika. selain itu akan di jelaskan pula perancangan *user* interface yang akan dibuat. Optimasi penjadwalan bimbingan skripsi dengan algoritme genetika terdiri dari beberapa proses sebagai berikut:

1. Inisialisasi parameter awal seperti Jumlah populasi, *cr (crossover rate)*, *mr (mutation rate)* dan Jumlah generasi.
2. Membuat populasi awal sesuai dengan jumlah populasi yang dimasukkan.
3. Membuat populasi baru menggunakan *crossover* dan mutasi untuk pembentukan *offspring*. Hasilnya nanti akan diikutkan bersama dengan *parent* dalam proses seleksi.
4. Menghitung nilai *fitness* masing-masing individu lalu di seleksi.
5. Proses seleksi dengan metode *elitism* untuk mendapatkan individu yang bertahan pada generasi selanjutnya.

Tabel 4.1 Aturan Penjadwalan Bimbingan Skripsi

No	***Constraint***
1	Terdapat bentrok antara jadwal bimbingan dengan jadwal mengajar dosen 1
2	Terdapat bentrok antara jadwal bimbingan dengan jadwal mengajar dosen 2
3	Terdapat bentrok antara jadwal mahasiswa dengan jadwal bimbingan yang sudah di tetapkan.
4	Mahasiswa yang memiliki dosen pembimbing sama (1 atau 2) tidak boleh bentrok dengan jadwal bimbingan mahasiswa satu dengan yang lain.

Dalam pembuatan jadwal bimbingan skripsi di FILKOM UB sistem diharapkan dapat memberikan solusi yang optimal dengan menggunakan algoritme genetika dilihat dari rata-rata nilai *fitness* yang terbaik. Nilai *fitness* mewakili seberapa banyak pelanggaran yang dilanggar. Penelitian ini merepresentasikan jadwal ke dalam deretan kromosom. Kromosom memiliki panjang sebanyak 250.

Data yang dimasukkan kedalam sistem yaitu data dosen dan data bimbingan yang disimpan ke dalam *database* MYSQL. Pengguna juga memasukan nilai parameter genetika seperti *popsize* (jumlah populasi) , *cr (crossover rate)*, *mr (mutation rate)* dan jumlah generasi. Keluaran dari sistem yaitu berupa jadwal bimbingan skripsi.

Tabel 4.2 Data Sampel Dosen Pembimbing Tahun Ajaran 2016/2017

Nama Mahasiswa	Dosen Pembimbing Satu	Dosen Pembimbing Dua
Muhammad Ali Aras R	Dany Primanita Kartikasari	Kasyful Amron
Fiki Nurhadiyanto	Eko Sakti P.	Kasyful Amron
Kukuh Wiliam M	Yuita Arum Sari	Achmad Arwan
Daneswara Jauhari	Imam Cholissodin	Candra Dewi
Fadhilla P Cahyani	M. Tanzil Furqon	Bayu Rahayudi
Rich Juniadi D S	Edy Santoso	Candra Dewi
Anang Hanafi	Randy Cahya W	Putra Pandu Adikara
Winda Cahyaningrum	Randy Cahya W	Agus Wahyu Widodo
Dwi Novi Setiawan	Candra Dewi	Sigit Adinugroho
Daffarez Elguska	Reza Andria Siregar	Rakhmadhany Primananda
Arief Sukma I	Rakhmadhany Primananda	Kasyful Amron
Ferdy Wahyurianto	Issa Arwani	Ratih Kartika Dewi
Edgar Juvianno S	Rakhmadhany Primananda	Kasyful Amron
Putri Rizqia Hardein	Rakhmadhany Primananda	Achmad Basuki

Tabel 4.3 Data Sampel Dosen Tahun Ajaran 2016/2017

Kode	L/P	Nama Dosen
1	L	Adam Hendra B, S.Kom., M.T., M.Sc.
2	P	Indriati, S.T., M.Kom.
3	L	Suprapto, S.T., M.T.
4	L	Denny Sagita R, S.Kom., M.Kom.
5	L	Edy Santoso, S.Si., M.Kom.
6	L	Tri Astoto K, S.T., M.T., Ph.D.
7	P	Ratih Kartika D, S.T., M.Kom.
8	L	Komang Candra B, S.Kom., M.T., M.Sc.
9	L	Agi Putra K, S.T, M.T.
10	L	Sabriansyah Rizqika A, S.T., M.Eng.

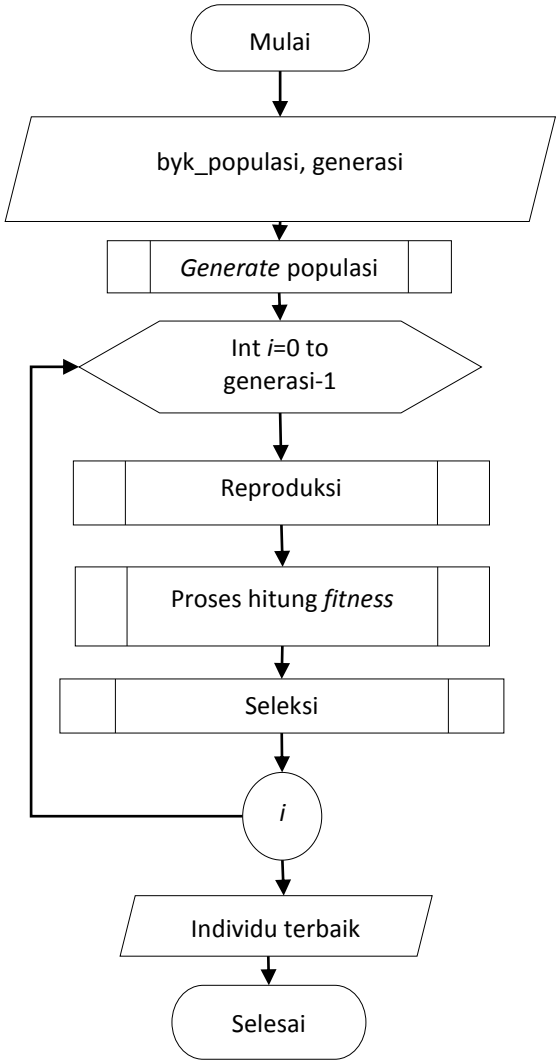
Tabel 4.4 Kode Jadwal Perkuliahan

Kode bimbingan	Keterangan
1	Senin , 07.00-08.40
2	Selasa, 07.00-08.40
3	Rabu, 07.00-08.40
4	Kamis, 07.00-08.40
5	Jumat, 07.00-08.40
6	Senin, 08.40-09.30
7	Selasa, 08.40-09.30
8	Rabu, 08.40-09.30
9	Kamis, 08.40-09.30
10	Jumat, 08.40-09.30
11	Senin, 09.30-10.20
12	Selasa, 09.30-10.20
13	Rabu, 09.30-10.20
14	Kamis, 09.30-10.20
15	Jumat, 09.30-10.20
16	Senin , 10.20-11.10
17	Selasa, 10.20-11.10
18	Rabu, 10.20-11.10
19	Kamis, 10.20-11.10
20	Jumat, 10.20-11.10
21	Senin, 11.10-12.00
22	Selasa, 11.10-12.00
23	Rabu, 11.10-12.00
24	Kamis, 11.10-12.00
25	Jumat, 11.10-12.00
26	Senin, 12.50-13.40
27	Selasa, 12.50-13.40
28	Rabu, 12.50-13.40
29	Kamis, 12.50-13.40
30	Jumat, 12.50-13.40
31	Senin, 13.40-14.30
32	Selasa, 13.40-14.30
33	Rabu, 13.40-14.30
34	Kamis, 13.40-14.30
35	Jumat, 13.40-14.30
36	Senin, 14.30-15.20
37	Selasa, 14.30-15.20
38	Rabu, 14.30-15.20
39	Kamis, 14.30-15.20
40	Jumat,14.30-15.20
41	Senin, 15.20-16.09
42	Selasa, 15.20-16.09
43	Rabu, 15.20-16.09
44	Kamis, 15.20-16.09
45	Jumat, 15.20-16.09
46	Senin, 16.10-17.10
47	Selasa, 16.10-17.10
48	Rabu, 16.10-17.10
49	Kamis, 16.10-17.10
50	Jumat, 16.10-17.10

4.2 Siklus Algoritme Genetika

Siklus algoritme genetika pada proses perancangan sistem adalah untuk mendapatkan solusi optimal. Langkah awal proses ini adalah peng*input*an jadwal pilihan mahasiswa untuk melakukan P0, serta nama dosen pembimbing 1 dan dosen pembimbing 2. Langkah selanjutnya adalah melakukan inialisasi populasi awal. Representasi kromosom didapat dari susunan pemilihan jadwal P0 yang dipilih.

Populasi awal dibangkitkan dari nilai parameter yang di masukan. *Offspring* dihasilkan dari jumlah kombinasi 2 induk. Kemudian individu dari populasi awal, *offspring* hasil dari reproduksi (*crossover* dan mutasi) digabungkan untuk proses seleksi. Pada proses seleksi dilakukan perhitungan nilai *fitness* pada setiap individu yang ada. Nilai *fitness* di dapat berdasarkan dari nilai pinalti. Proses penyelesaian masalah optimasi penjadwalan bimbingan skripsi khusus menggunakan algoritme genetika dapat dilihat pada Gambar 4.1.

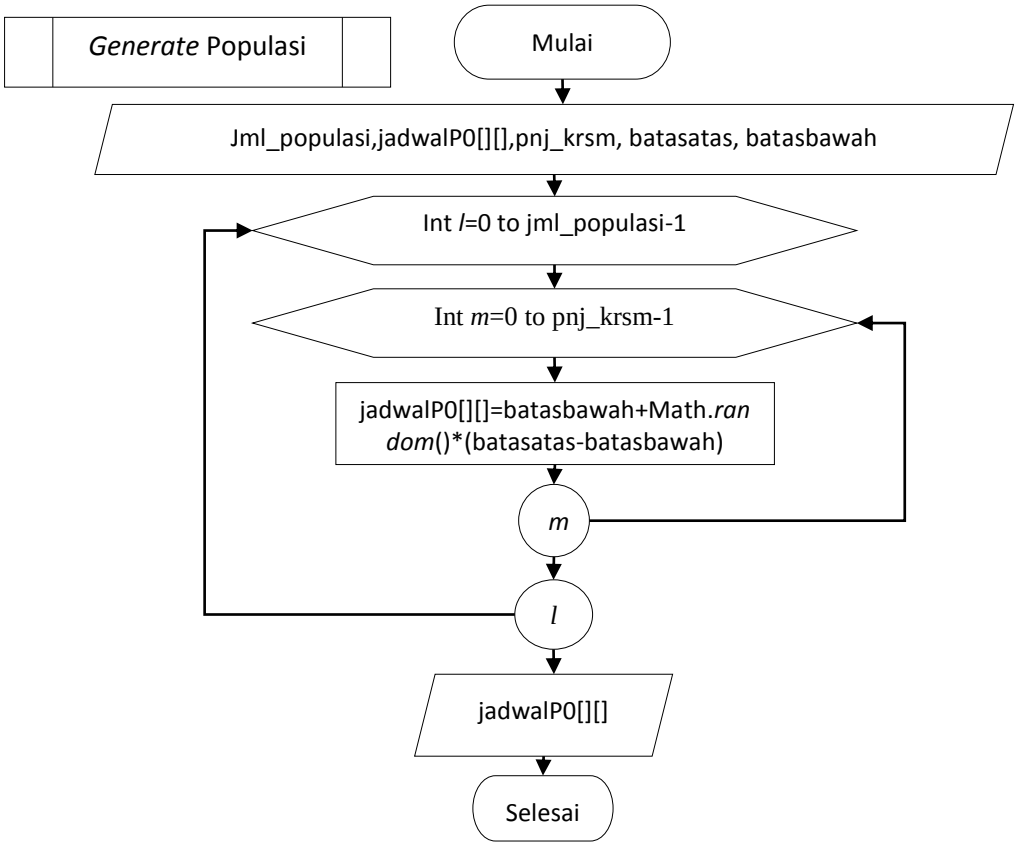


Gambar 4.0.1 Diagram Alir Proses Pembentukan Susunan Kromosom Optimal

Seperti pada Gambar diatas terdapat proses utama untuk menyelesaikan permasalahan optimasi penjadwalan bimbingan skripsi yaitu proses *generate* populasi, reproduksi, proses hitung *fitness* dan seleksi. Proses kerjanya yaitu setelah memasukkan data bimbingan, nama dosen pembimbing 1 dan nama doen pembimbing 2 serta nama mahasiswa maka sistem akan membuat populasi awal sesuai dengan jumlah populasi dan parameter algoritme genetika yang dimasukkan oleh pengguna. Selanjutnya akan dilakukan proses ulang untuk membentuk *offspring* dengan melakukan proses reproduksi (*crossover* dan mutasi). Hasil *offspring* berdasarkan proses reproduksi tersebut akan dilakukan perhitungan nilai *fitness* dari setiap individu yang ada, kemudian dilakukan proses seleksi.

4.2.1 *Generate* Populasi Awal

Pada proses ini sistem akan membuat populasi awal secara acak yang terdiri dari sejumlah individu yang dimasukan. Proses pembuatan populasi ini dapat dilihat pada Gambar 4.2 berikut:



Gambar 4.0.2 Diagram Alir Proses *Generate* Populasi Awal

Berdasarkan Gambar 4.2 dapat di jelaskan langkah-langkah dalam pembuatan populasi awal yaitu seperti di bawah ini:

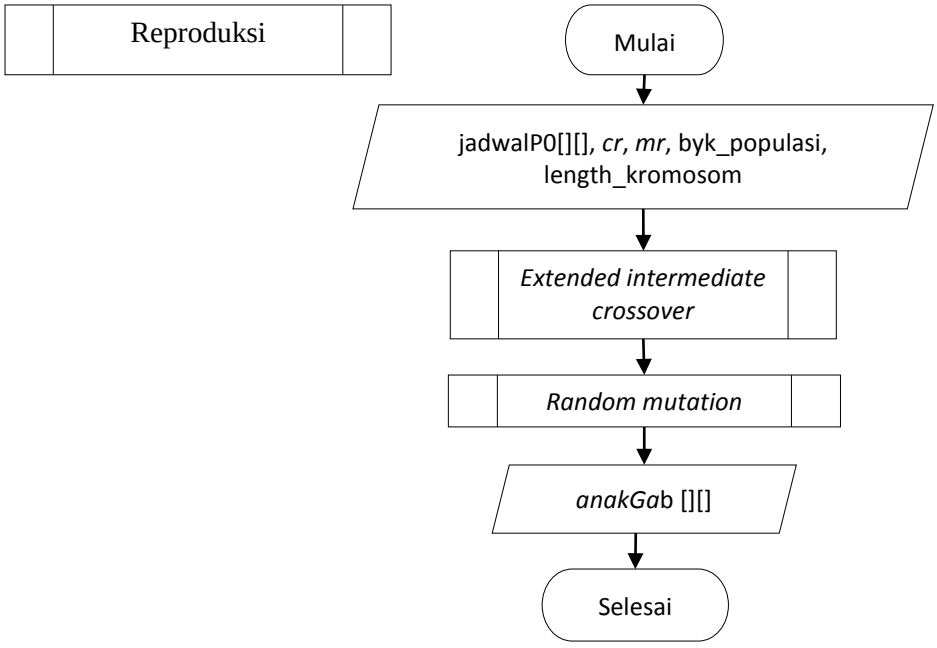
1. Sistem menerima masukan berupa jumlah populasi (*popsize*) yang akan di optimasi oleh sistem,pnj_krsm (panjang kromosm) dan jadwalP0[][] yang

akan membangkitkan individu dengan jumlah populasi yang ditentukan secara *random*.

2. Banyaknya jumlah populasi yang ditentukan oleh pengguna akan berpengaruh terhadap banyaknya proses yang akan di lakukan oleh sistem.
3. Hasil *output* ini berupa populasi yang mana digunakan untuk proses selanjutnya.

4.2.2 Reproduksi

Pada proses ini sistem akan melakukan pembuatan *offspring* menggunakan *crossover* dan mutasi. Hasil *crossover* dan mutasi nanti akan di ikutakan bersama *parent* pada proses seleksi. Pada proses *crossover* disini menggunakan *extended intermediate crossover* sedangkan pada mutasi menggunakan *random mutation.* Langkah-lagkah yang digunakan pada proses reproduksi dapat dilihat pada Gambar 4.3 berikut:



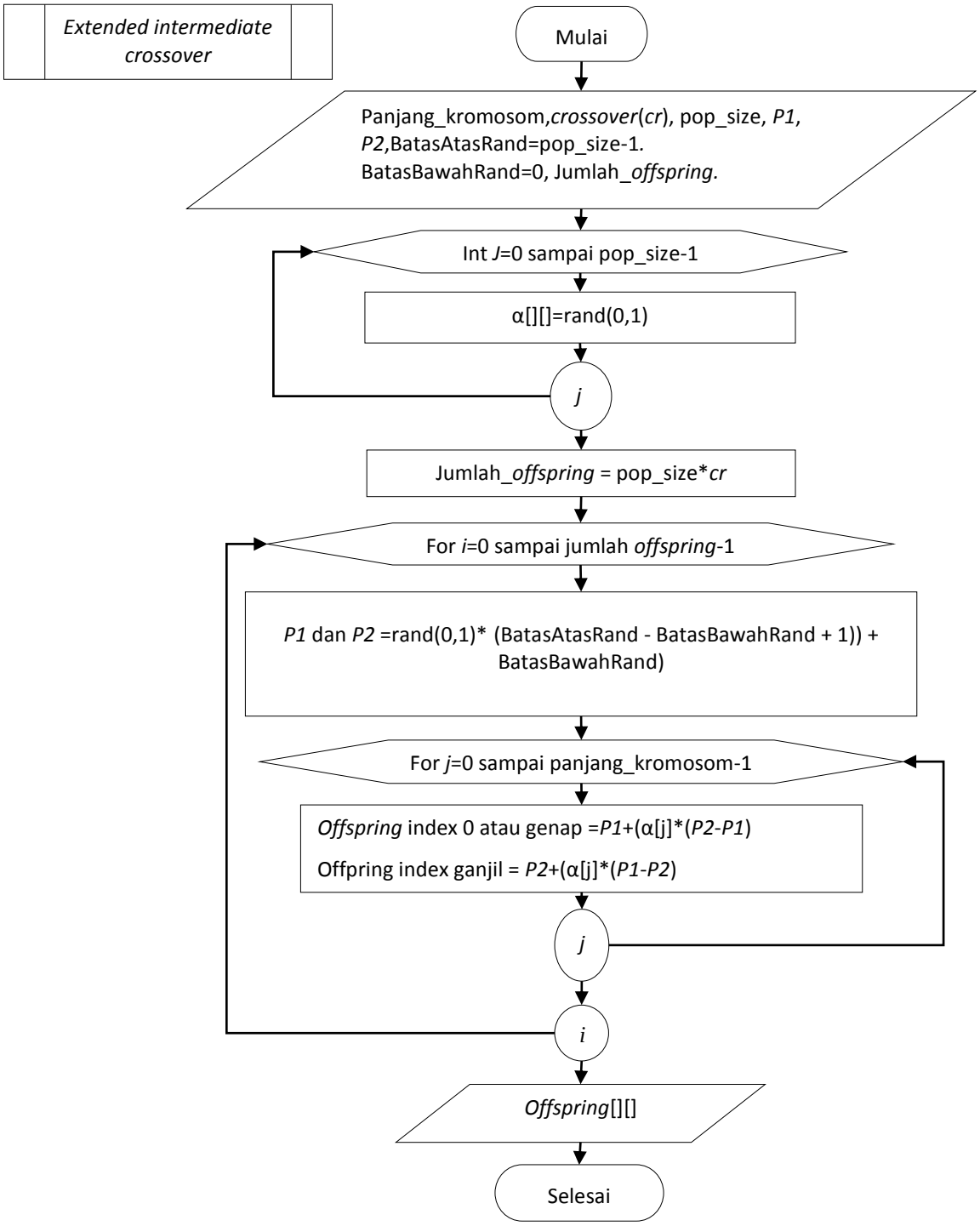
Gambar 4.0.3 Diagram Alir Proses *Crossover*

Berdasarkan diagram alur diatas dapat dijelaskan langkah-langkah dalam proses *crossover* yaitu seperti di bawah ini:

1. Berdasarkan masukan data berupa jadwalP0[][] yaitu individu yang sudah dilakukan penginialisasian nilai kromosom, serta parameter dari algoritme genetika seperti nilai *crossover* (*cr*), nilai *mutation* (*mr*), byk_populasi (*popsize*) dan panjang kromosom (length_kromosom).
2. Sistem akan melakukan proses *crossover*.
3. Sistem akan melakukan proses mutasi.

4.2.2.1 Proses *Exteded Intermediate Crossover*

Untuk mendapatkan kromosom yang baru dapat dilaukan dengan operasi penyilangan antara dua kromosom induk yang dipilih secara acak, inilah yang disebut proses *crossover*. Gambar 4.4 adalah alir proses *extended intermediate crossover*.



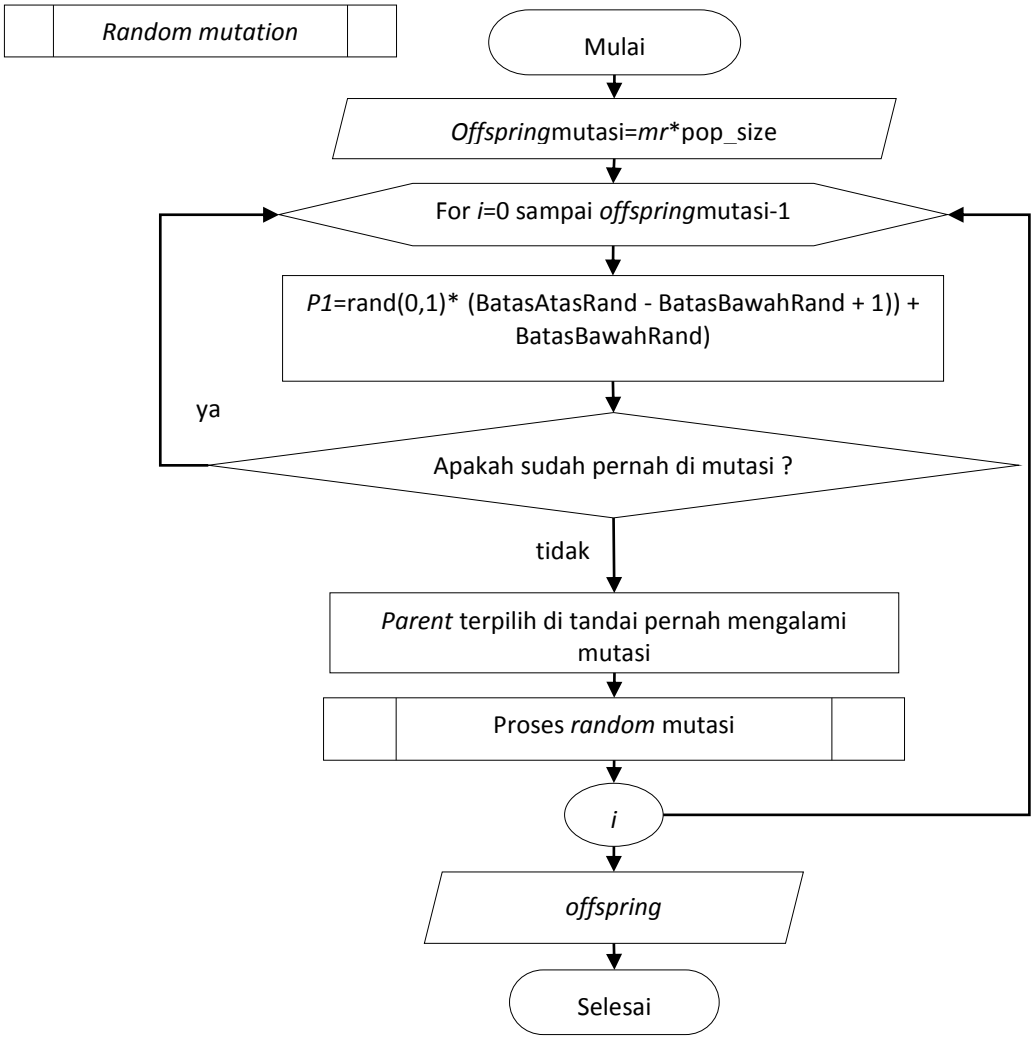
Gambar 4.0.4 Diagram Alir Proses *Extended Intermediate Crossover*

Langkah pada proses *extended intermediate crossover* seperti pada Gambar 4.4 yaitu seperti di bawah ini:

1. Sistem menerima masukan berupa *offspring_cr* (jumlah anak dari proses *crossover*), banyak populasi (*popsize*) dan panjang kromosom.
2. Menentukan bilangan *random* α , *parent* (*P1* dan *P2)*.
3. Menghitung nilai *offspring* berdasarkan rumus *extended intermediate crossover.*
4. Sistem akan memberikn keluaran berupa hasil *offspring* dari hasil proses *crossover*.

4.2.2.2 Proses *Random mutation*

Untuk mendapatkan gen yang lebih baik dari induk, maka dapat dilakukan perubahan gen (satu atau beberapa gen) pada proses ini. Gambar 4.5 menunjukan diagram alir proses *random mutation.*



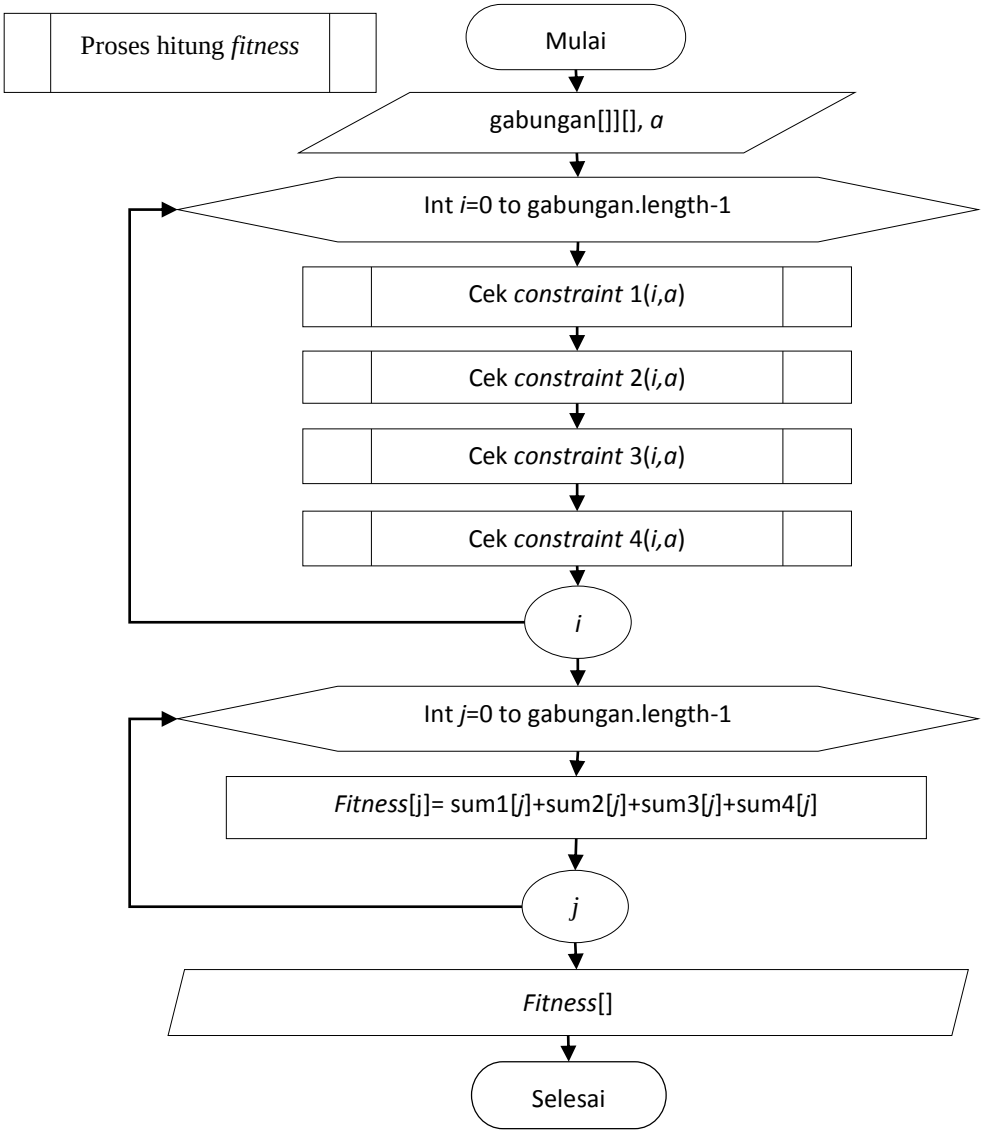
Gambar 4.0.5 Diagram Alir Proses *Random mutation*

Langkah pada proses *random mutation* seperti pada Gambar 4.5 yaitu seperti di bawah ini:

1. Menentukan banyaknya jumlah *offspring mr* dengan perhitungan *mr*popsize*.
2. Menentukan secara acak *parent* yang akan di mutasi.
3. Lalu di proses dengan perhitungan *random mutation*.
4. Setelah itu sistem akan mengeluarkan hasil dari proses *random mutation.*

4.2.3 Proses Evaluasi *Fitness*

Untuk mengukur baik tidaknya suatu individu yang ada, dapat diukur dengan melihat nilai *fitness* dari setiap individu yang didapatkan, semakin besar nilai *fitness* maka akan semakin baik. Gambar 4.6 menunjukan diagram alir proses evaluasi *fitness*.



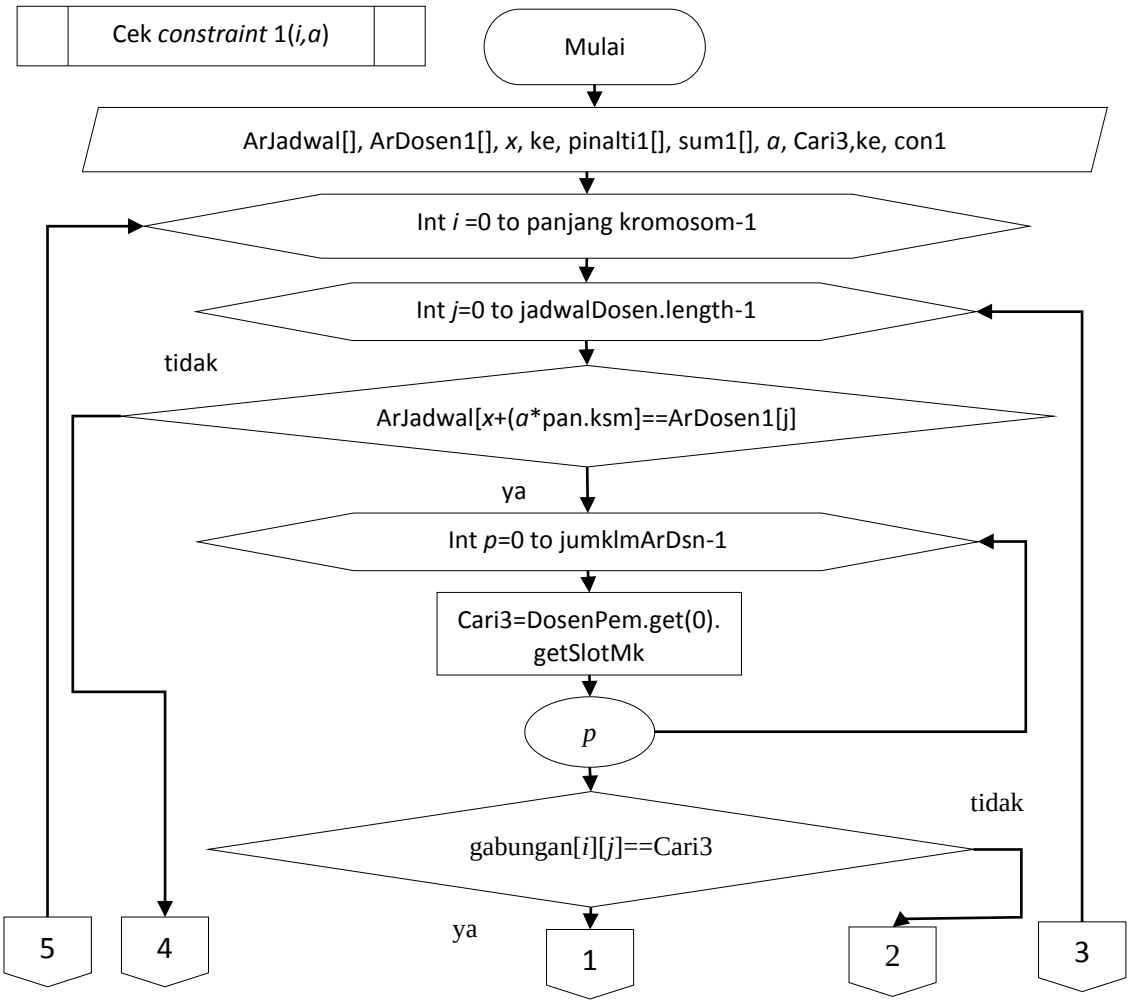
Gambar 4.0.6 Diagram Alir Proses Perhitungan *Fitness*

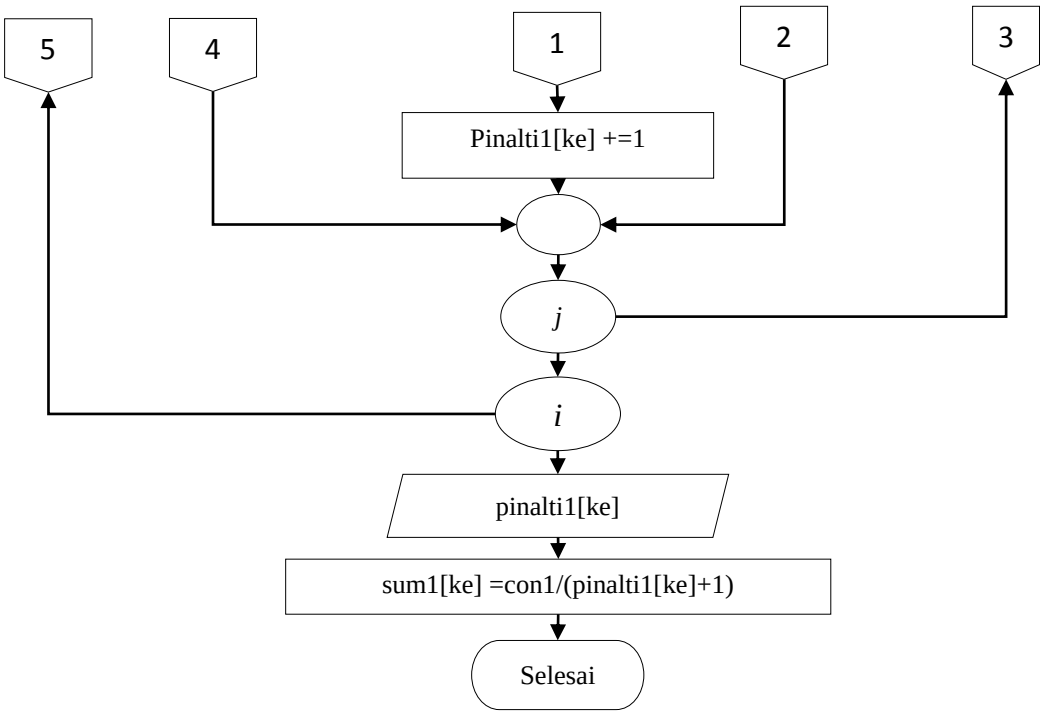
Langkah pada proses perhitungan nilai *fitness* seperti pada Gambar 4.6 yaitu seperti di bawah ini:

1. Sistem akan menerima masukan berupa gabungan (individu yang sudah digabung antara induk dan anak), dan *a* (nilai *varia*bel *a*).
2. Perulangan *i* sampai panjang gabungan-1 dilakukan pada nomor 3 sampai 6.
3. Sistem akan mengecek prosedur cek *hard constraint* 1.
4. Sistem akan mengecek prosedur cek *hard constraint* 2.
5. Sistem akan mengecek prosedur cek *hard constraint* 3.
6. Sistem akan mengecek prosedur cek *hard constraint* 4.
7. Sistem akan mengulang iterasi 1 sampai jumlah gabungan dan memproses perhitungan *fitness* lalu memberikan hasil keluaran berupa nilai *fitness* setiap individu gabungan.

4.2.3.1 Proses Cek *Constraint* 1

Cek *hard constraint* 1 yaitu mengecek apakah terdapat bentrok jadwal bimbingan dengan jadwal mengajar dosen. Gambar 4.7 menunjukan diagram alir proses *hard constraint* 1.





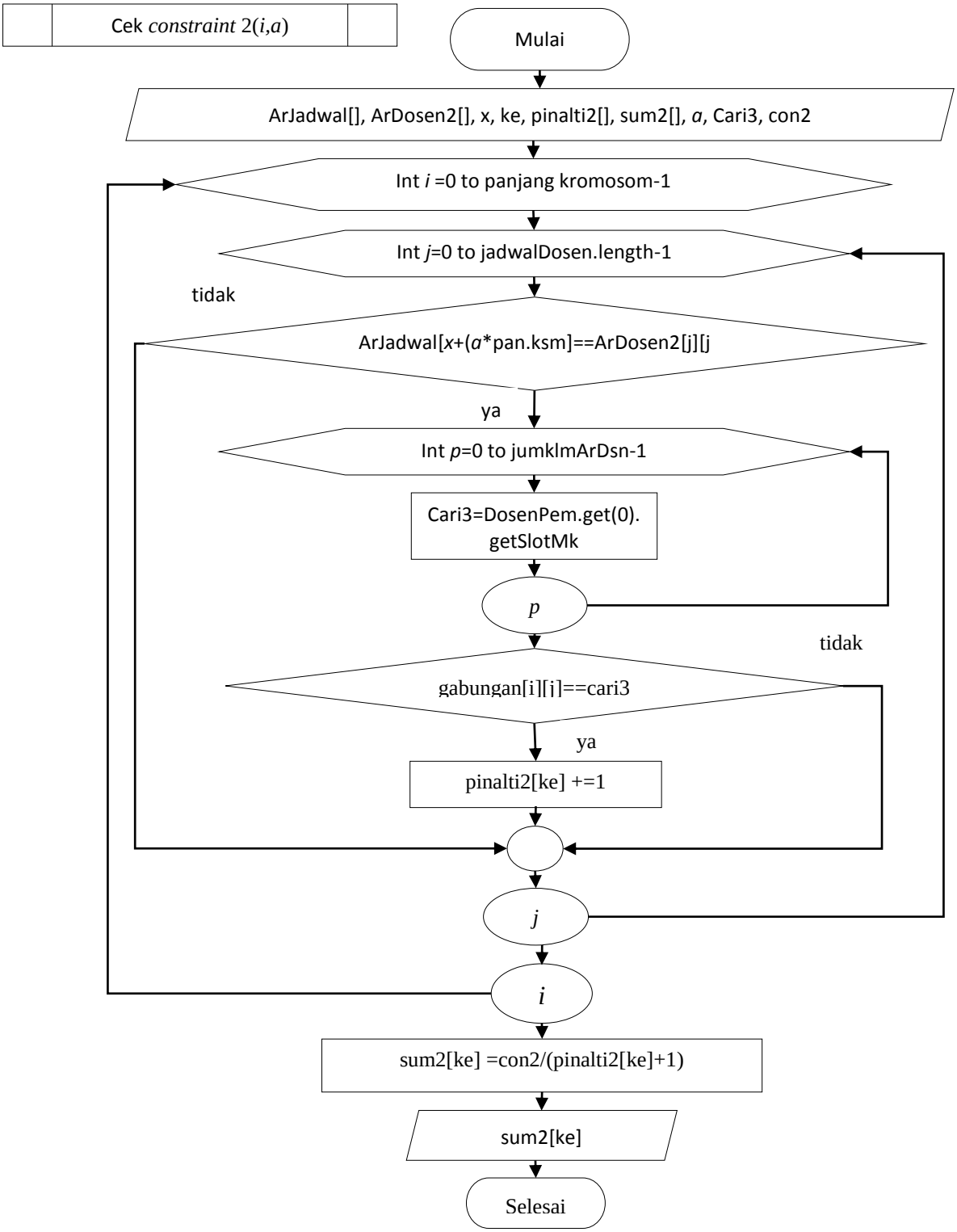
Gambar 4.0.7 Diagram Alir Proses Cek *Constraint* 1

Langkah pada proses *extended intermediate crossover* seperti pada Gambar 4.7 yaitu seperti di bawah ini:

1. Sistem akan menerima masukan ArDosen[] (jadwal mengajar dosen), ArJadwal (jadwal perkuliahan mahasiswa), ke (individu setiap populasi), sum1[] (penampungan nilai dari jumlah pinalti1), Cari3, *a*, dan pinalti1[] (jumlah nilai bobot dari pelanggaran yang dilanggar pada *constraint* 1), con1 (nilai konstanta bobot *hard constraint* 1).
2. Perulangan *i* sampai dengan panjang kromosom.length-1, Perulangan *j* sampai jadwalDosen.length-1.
3. Kondisi jika jadwal dosen1 dan jadwal mahasiswa sama .
4. Perulangan *i* sampai dengan jumlah kolom pada ArDosen-1.
5. Sistem akan memproses cari3 sama dengan jadwal dosen yang didapat pada DosenPem pada baris ke 0 yang bernama getSlotMK.
6. Pengulangan untuk *p* sampai dengan jumlah kolom jadwal dosen yaitu 12. Kondisi Jika gabungan yang dicari ternyata sama dengan jadwal dosenpembimbing ke 1 (cari3). Jika sama maka pinalti akan bertambah.
7. Sistem akan menghasilkan keluaran berupa bobot pinalti pada *constraint* 1.
8. Penjumlahan nilai pinalti dengan rumus yang ada di flowchart dan dimasukkan kedalam *array* sum1.

4.2.3.2 Proses Cek *Constraint* 2

Proses cek *hard constraint* 2 yaitu mengecek apakah terdapat bentrok antara jadwal bimbingan dengan jadwal mengajar dosen 2. Gambar 4.8 menunjukan diagram alir proses *hard constraint* 2.



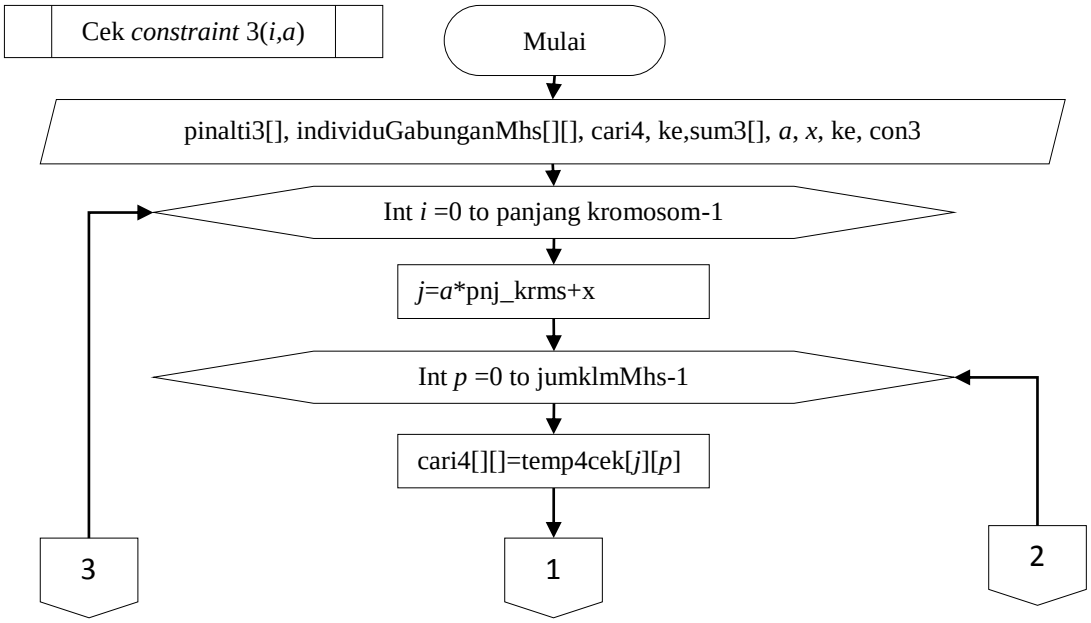
Gambar 4.0.8 Diagram Alir Proses Cek *Constraint* 2

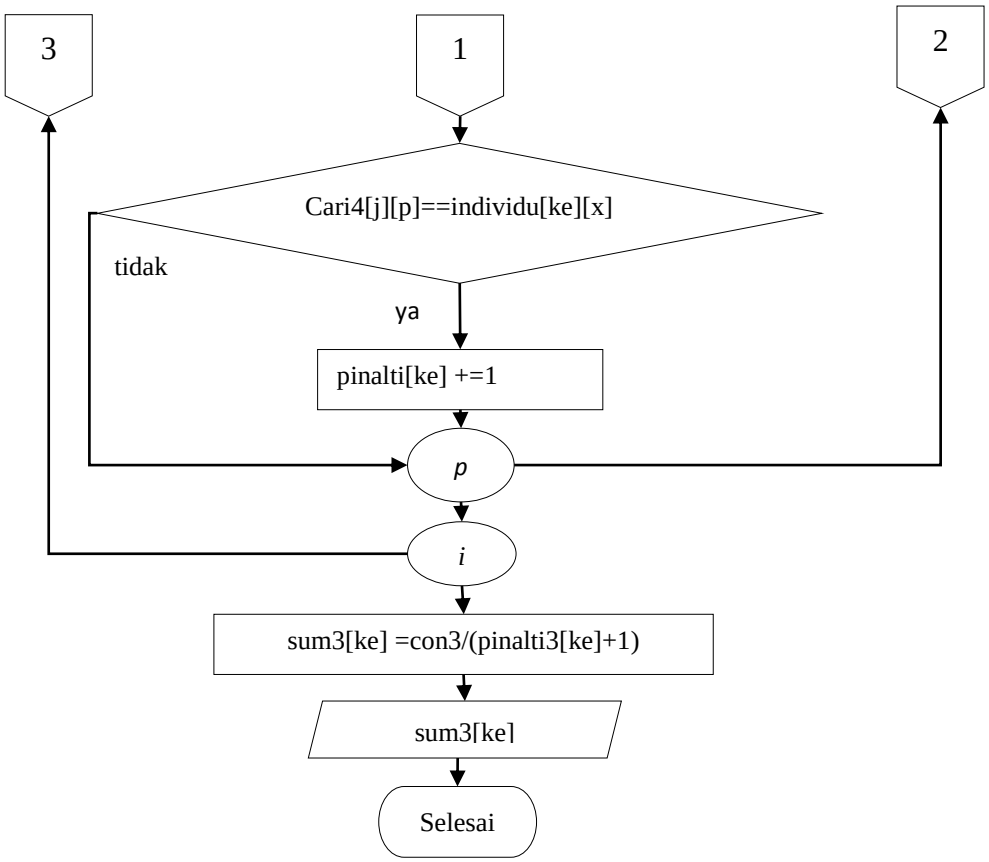
Langkah pada proses *extended intermediate crossover* seperti pada Gambar 4.8 yaitu seperti di bawah ini:

1. Sistem akan menerima masukan ArDosen2[] (jadwal mengajar dosen pembimbing yang ke-2), ArJadwal (jadwal perkuliahan mahasiswa), ke (individu setiap populasi), sum2[] (penampungan nilai dari jumlah pinalti), Cari3, *a*, dan pinalti2[] (jumlah nilai bobot dari pelanggaran yang dilanggar pada *constraint* 2), con2 (nilai konstanta bobot *hard constraint* 2).
2. Perulangan *i* sampai dengan panjang kromosom.length-1, Perulangan *j* sampai jadwalDosen.length-1.
3. Kondisi jika jadwal dosen2 dan jadwal mahasiswa sama .
4. Perulangan *i* sampai dengan jumlah kolom pada ArDosen-1.
5. Sistem akan memproses cari3 sama dengan jadwal dosen yang didapat pada DosenPem pada baris ke 0 yang bernama getSlotMK.
6. Jika kondisi gabungan yang dicari ternyata sama dengan jadwal dosenpembimbing ke 2 (cari3).
7. Jika sama maka pinalti akan bertambah.
8. Sistem akan menghasilkan keluaran berupa bobot pinalti pada. *constraint* 1.
9. Penjumlahan nilai pinalti dengan rumus yang ada di flowchart dan dimasukkan kedalam *array* sum2.

4.2.3.3 Proses Cek *Constraint* 3

Proses cek *hard constraint* 3 adalah pengecekkan apakah terdapat bentrok antara jadwal mahasiswa dengan jadwal bimbingan yang sudah di tetapkan.. Gambar 4.9 menunjukan diagram alir proses *hard constraint* 3.





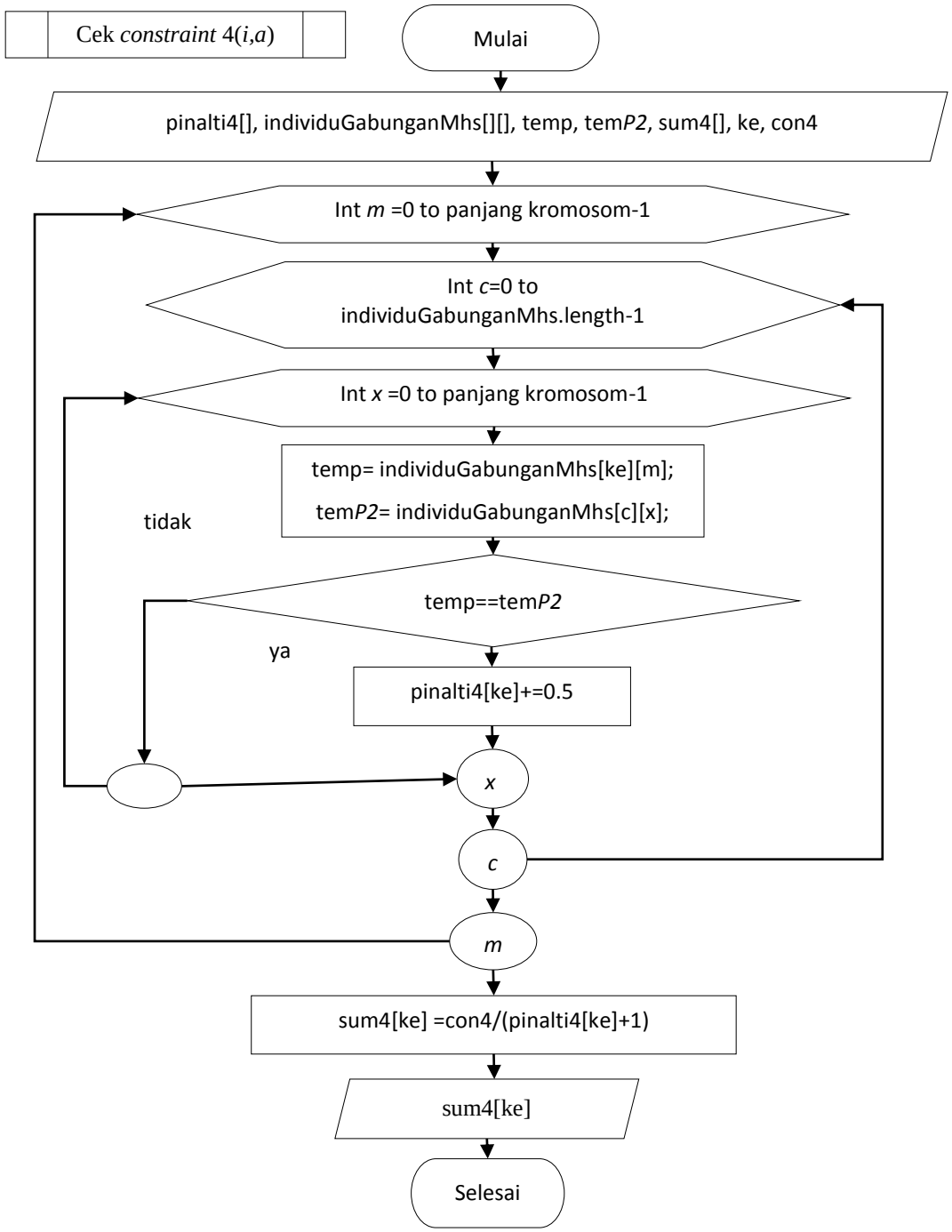
Gambar 4.0.9 Diagram Alir Proses Cek *Constraint* 3

Langkah pada proses *extended intermediate crossover* seperti pada Gambar 4.9 yaitu seperti di bawah ini:

1. Sistem akan menerima masukan pinalti3[] (jumlah pelanggaran yang dilanggar pada *constraint* 3), individuGabunganMhs[][] (individu dari induk dan anak yang telah digabung), cari4, dan *ke,* con3 (nilai konstanta bobot *hard constraint* ke-3).
2. Perulangan *i* sampai dengan panjang kromosom-1.
3. Deklarasi nilai *j*.
4. Perulangan *p* sampai dengan jumlah kolom pada mahasiswa-1.
5. Deklarasi nilai cari4.
6. Kondisi jika cari4 dan jadwal bimbingan sama maka pinalti[ke] ditambahkan 1.
7. Sistem akan menghasilkan keluaran berupa bobot pinalti pada *constraint* 3.
8. Sistem akan menjumlahkan nilai banyaknya jumlah pinalti pada rumus di flowchart yang akan di masukkan kedalam *array* sum3.

4.2.3.4 Proses Cek *Constraint* 4

Proses cek *hard constraint* 4 adalah pengecekkan apakah mahasiswa yang memiliki dosen pembimbing sama (1 atau 2) bentrok dengan jadwal bimbingan mahasiswa satu dengan yang lain. Gambar 4.10 menunjukan diagram alir proses *hard constraint* 4.



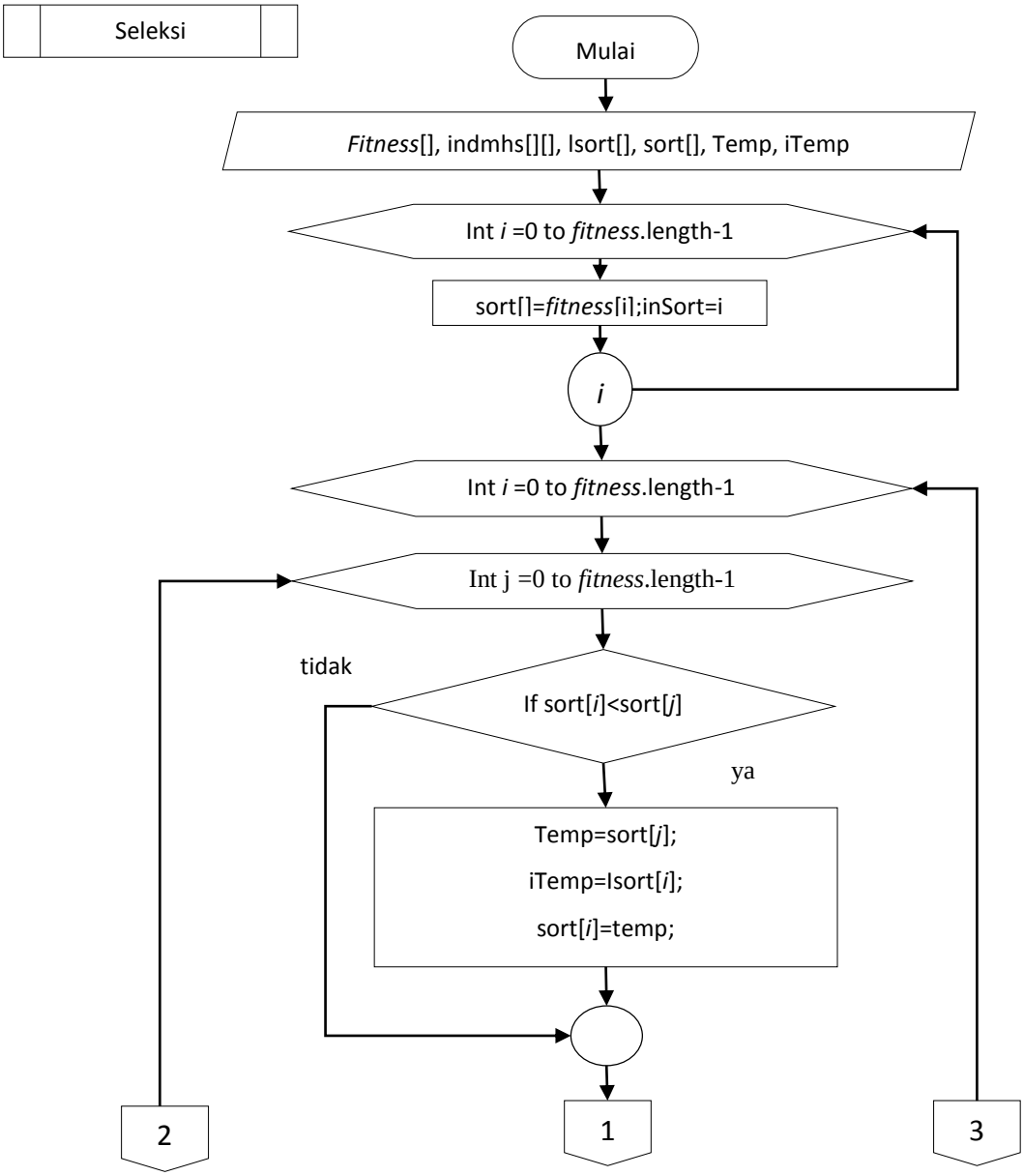
Gambar 4.0.10 Diagram Alir Proses Cek *Constraint* 4

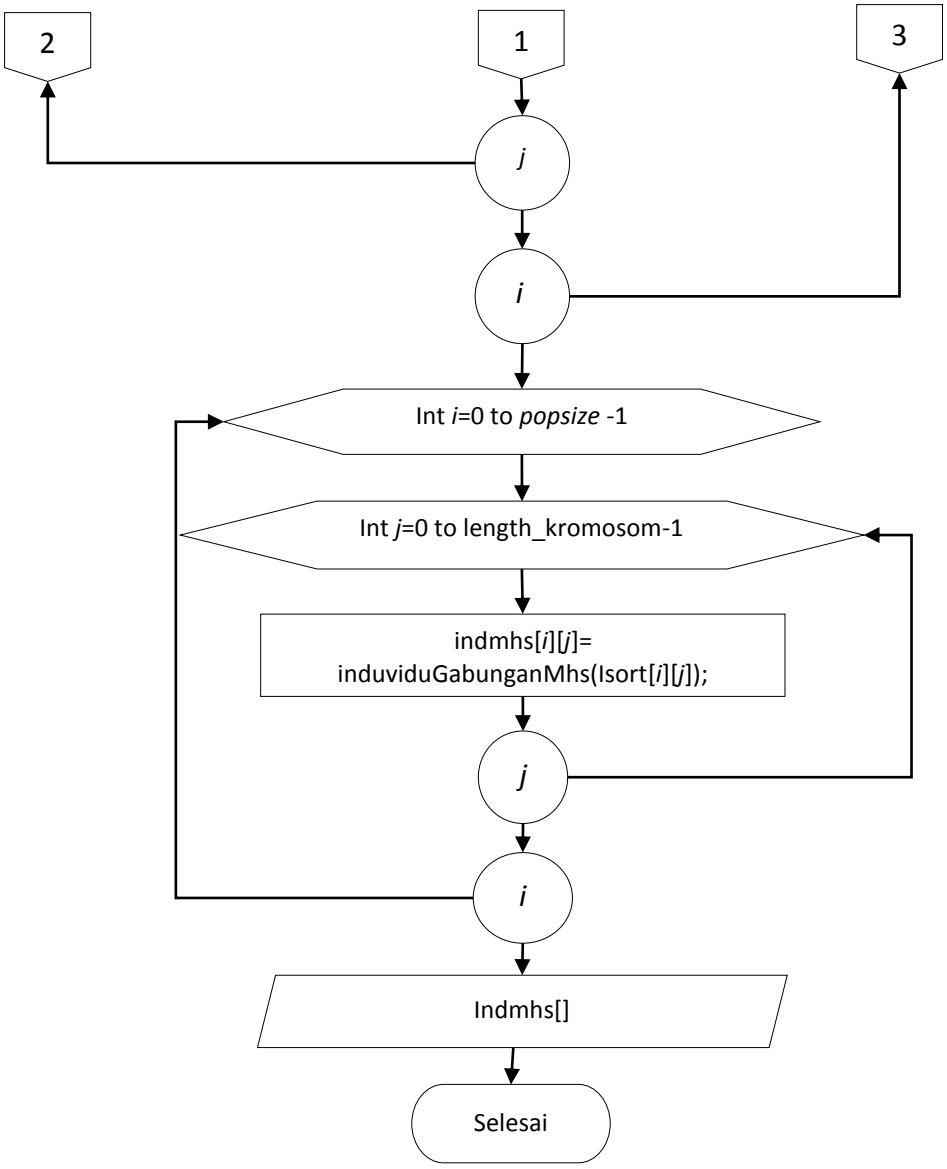
Langkah-langkah pada proses *extended intermediate crossover* seperti pada Gambar 4.10 yaitu seperti di bawah ini:

1. Sistem akan menerima masukan individuGabunganMhs, pinalti4[], temp, tem*P2*, sum4[], con4 (nilai konstanta bobot *soft constraint*)
2. Perulangan *m* sampai dengan panjang kromosom.length-1, Perulangan *c* sampai jadwalDosen.length-1.
3. Jika temp memiliki nilai yang sama dengan tem*P2* maka nilai pinalti bertambah 0,5.
4. Sistem akan menghasilkan keluaran berupa bobot pinalti pada *constraint* 4.
5. Sistem akan menjumlahkan nilai pinalti kedalam *array* sum4.

4.2.4 Proses Seleksi

Pada tahap ini peneliti menggunakan metode *elitism*. Proses ini hanya mengambil sebanyak *popsize* dari populasi anak dan induk. Gambar 4.11 menunjukan diagram alir dari proses seleksi dengan metode seleksi *elitism*.





Gambar 4.0.11 Diagram Alir Proses Seleksi *Elitism*

Langkah-langkah seleksi *elitism* berdasarkan Gambar 4.11 adalah sebagai berikut:

1. masukkan berupa nilai *fitness[]* (nilai *fitness* yang didapat setelah melakukan proses algoritme genetika), indmhs[][] (individu sesuai dengan jumlah *popsize*), dan variabel Temp, iTemp, Isort, sort sebagai variabel pembantu saja.
2. Mengurutkan nilai *fitness* dari terbesar ke terkecil.
3. Pengambilan beberapa populasi berdasarkan nilai *fitness* tertinggi sebanyak jumlah *popsize*.
4. Sistem akan menghasilkan nilai keluaran berupa indmhs yaitu kromosom yang baru untuk dapat masuk ke proses selanjutnya.

4.3 Perhitungan Manualisasi Proses Algoritme Genetika

Perhitungan manualisasi ini dilakukan pada seluruh proses yang terdapat pada algoritme genetika yaitu inisialisasi kromosom, proses reproduksi, evaluasi dan seleksi yang nantinya akan mengeluarkan *output* berupa solusi yang optimal. Data jadwal yang dihitung sebanyak 15 jadwal yang akan merepresentasikan kromosom dengan panjang 5 kromosom. Beberapa parameter yang digunakan dalam perhitungan manual yang telah diinisialisasi terlebih dahulu yaitu:

- Banyaknya Generasi : 1
- *Popsize* : 3
- Nilai *cr* : 0,8
- Nilai *mr* : 0,3

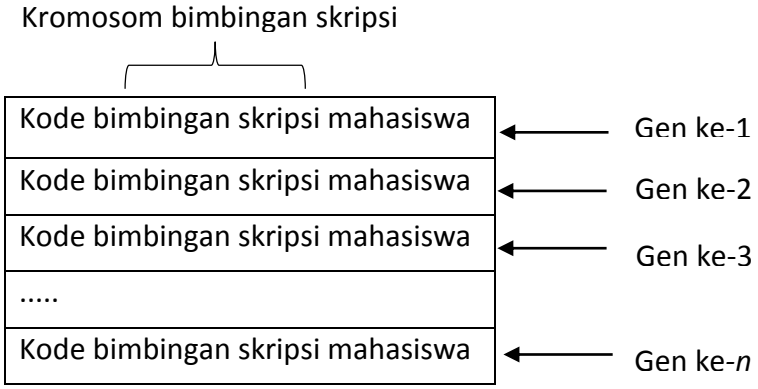
Penelitian ini memiliki beberapa batasan (*constraint*) yang nantinya akan berpengaruh terhadap nilai *fitness* jika batasan tersebut dilanggar maka nilai bobot akan bertambah. *Constraint* tersebut memiliki nilai bobot masing-masing. Pada *hard constraint*, bobot pinalti diverikan lebih besar dari bobot pada *soft constraint*. Pemberian nilai bobot pinalti telah dikonsultasikan dan mendapat persetujuan dari pakar. Pada Tabel 4.5 terdapat empat *constraint (*tiga *hard constraint* dan satu *soft constraint)* dan bobot pinalti penjadwalan bimbingan skripsi.

Tabel 4.5 *Constraint* dan Bobot Pinalti Penjadwalan Bimbingan Skripsi

No	*Constraint*	Nilai Bobot
1	Terdapat bentrok antara jadwal perkuliahan mahasiswa dengan jadwal mengajar dosen 1	1
2	Terdapat bentrok antara jadwal perkuliahan mahasiswa dengan jadwal mengajar dosen 2	1
3	Terdapat bentrok antara jadwal mahasiswa dengan jadwal bimbingan yang sudah di tetapkan.	1
4	Mahasiswa yang memiliki dosen pembimbing sama(1 atau 2) tidak boleh bentrok dengan jadwal bimbingan mahasiswa satu dengan yang lain.	0,5

4.3.1 Inisialisasi Kromosom

Ukuran populasi awal yang ditetapkan pada sampel data perhitungan ini yaitu 3 individu. Proses ini, kromosom di representasikan ke dalam bentuk *array* dan panjang kromosom yaitu 5. Gen ke-*n*, dimana n menunjukan slot bimbingan ke-*n*, pada gen ke-1 berisi jadwal bimbingan ke-1. Representasi kromosom ditunjukan pada Gambar 4.17.



Gambar 4.12 Representasi Kromosom

Langkah 1 : Membangkitkan nilai acak yang akan dimasukan kedalam gen pada interval 1-250. 250 mewakili bahwa setiap minggu terdapat 50 slot jadwal, dan terdapat 5 minggu untuk proses P0 jadi 50 x 5 = 250. Hasil inisialisasi terhadap 3 individu ditunjukan pada Tabel 4.6. Pada Tabel 4.7 adalah contoh dari representasi kromosom. Individu ke-1 pada gen ke-1 merepresentasikan kode jadwal bimbingan skripsi berdasarkan mahasiswa 1.

Tabel 4.6 Kode Slot Jadwal Pengecekan

Jam\Hari	senin	selasa	rabu	kamis	jumat	minggu
07.00-08.40	1	2	3	4	5	
08.40-09.30	6	7	8	9	10	
09.30-10.20	11	12	13	14	15	
10.20-11.10	16	17	18	19	20	
11.10-12.00	21	22	23	24	25	
Istirahat						1
12.50-13.40	26	27	28	29	30	
13.40-14.30	31	32	33	34	35	
14.30-15.20	36	37	38	39	40	
15.20-16.09	41	42	43	44	45	
16.10-17.10	46	47	48	49	50	

Tabel 4.7 Individu Pada Populasi Awal

Individu ke-*i* /Gen ke-*j*	Mahasiswa				
	1	2	3	4	5
1	178	137	123	142	133
2	48	125	51	124	83
3	167	120	6	139	144

4.3.2 Proses Reproduksi

Setelah proses inialisasi awal pada tabel kemudian dilakukan proses reproduksi. Reproduksi pada penelitian ini ada *crossover* dan mutasi dimana menggunakan *extended intermediate crossover* dan *random mutation*.

4.3.2.1 Proses *Crossover*

Pada proses ini adalah menghitung jumlah anak yang akan dihasilkan karena nilai *cr* =0,8 dan *popsize* 3, maka akan menghasilkan 2 anak (*offspring*) maka proses *crossover* dilakukan sebanyak 1 kali, karena satu kali proses *crossover* menghasilkan dua anak. Langkah-langkah dalam proses *crossover* menggunakan metode *extended intermediate crossover* yaitu:

Langkah 1: Memilih sebanyak dua individu yang akan dijadikan induk secara acak. Misalkan *P1* (sebagai *parent* 1) dan *P2* (sebagai *parent* 2).

P1 = [167, 120, 6, 139, 144]

P2= [48, 125, 51, 124, 83]

Langkah 2: Menghitung nilai dari *offspring* berdasarkan rumus pada Persamaan 2.1.

Langkah 3: menentukan nilai *a*. Nilai *a* adalah nilai range yang misalkan terdapat diantara nilai 0 sampai 1. Misalkan *a* yang didapat yaitu:

a = [0.,218, 0.08, 0.71, 0.113, 0.5952]

Tabel 4.8 Hasil Proses *Crossover*

Offspring	**Hasil Kromosom**
C1	[149, 136, 71, 139, 103]
C2	[76, 125, 102, 126, 112]
C4	[73, 124, 18, 125, 119]
C5	[141, 120, 1, 137, 107]

4.3.2.2 Proses Mutasi

Pada proses ini yaitu menghitung banyak anak untuk hasil mutasi, karena nilai *mr* yaitu 0,3 dan *popsize* 3, maka 3 x 0,3 =0,9 di bulatkan menjadi 1. *Offspring* yang dihasilkan sebanyak 1. Mutasi akan di lakukan satu kali. Untuk pemilihan individu yang akan menjadi *parent* dilakukan secara *random*, misalkan terpilih individu *P1*:

P1 = [178, 137, 123, 142, 133]

Untuk proses selanjutnya yaitu menentukan nilai *random* r, berdasarkan rumus *random mutation* yang terdiri antara 0 sampai 1:

r = [0.218 ,0,08 , 0.71 , 0,113 , 0,59]

rumus *random mutation* sendiri yaitu:

Xi'= Xn' + r (max-min)

Keterangan :

- (maxi-mini) = domain variabel *ij*.
- Xi',Xn' = *offspring* yang dihasilkan.
- *r* = ranger r (misalkan bilangan *random* interval [0..1]

Jika yang terpilih adalah index ke 0, max 178, dan min 48. pada gabungan invdividu awal dengan hasil *crossover* didapatkan nilai max dan min yang akan digunakan untuk proses *random mutation*.

Tabel 4.9 Hasil Proses Mutasi

Offspring	**Kromosom**
C3	[206, 137,123,142,133]

4.3.3 Proses Perhitungan Evaluasi *Fitness*

Pengecekkan nilai *fitness* berdasarkan *constraint* dari penjadwalan bimbingan skripsi berdasarkan pada Tabel 4.1 pada setiap individu. Sebelum masuk ke pengecekan *constraint*, dari populasi awal sampai anak dilakukan kondisi pengurangan dikarenakan jadwal mengajar dosen dan jadwal perkuliahan mahasiswa adalah tetap dan kode dari jadwal tersebut yaitu 1 sampai 50. Setelah selesai dilakukan pengecekan semua *constraint* kemudian menghitung nilai *fitness* masing-masing individu.

Tabel 4.10 Perubahan Kromosom untuk Pengecekan

Offspring	**Hasil Kromosom**
P1	[178, 137, 123, 142, 133]
P2	[48, 125, 51, 124, 83]
P3	[167, 120, 6, 139, 144]
C1	[149, 136, 71, 139, 103]
C2	[76, 125, 102, 126, 112]
C4	[73, 124, 18, 125, 119]
C5	[141, 120, 1, 137, 107]

→

Offspring	**Hasil Kromosom**
P1	[28, 37, 23, 42, 33]
P2	[48, 25, 1, 24, 53]
P3	[17, 20, 6, 39, 44]
C1	[49, 36, 71, 39, 3]
C2	[26, 25, 2, 26, 12]
C4	[23, 24, 18, 25, 19]
C5	[41, 20, 1, 37, 7]

Pada Tabel 4.10 pada P1 dengan gen sebesar 178 akan di kurang dengan 150 maka akan didapatkan gen sebesar 28. Disini terdapat beberapa kondisi pertama, jika gen tersebut lebih dari 50 dan lebih kecil dari 100 maka akan dikurang 50. Kedua jika gen tersebut lebih dari 100 dan kurang dari 150 maka akan dikurang 100. Ketiga, jika gen tersebut lebih dari 150 dan kurang dari 200 maka akan dikurang 150. Keempat, jika gen tersebut lebih dari 200 dan kurang dari 250

maka akan dikurang 200. Kelima jika kurang sama dengan dari 50 maka nilai akan tetap.

Tabel 4.11 Sample Nama Dosen dan Kode Mengajar

Nama Dosen	Kode Mengajar
Tanzil Furqon	5,7,8,10,11,12,13,16,17,19,21,24,26,28,31,33,38,43,48
Wayan Firdaus	2,3,4,7,8,9,10,15,20
Adam Hendra	4,9,14,26,27,31,32,36,37,41,42,43,46,47,48
Bayu Rahayudi	8,12,13,17,18,22,23,29,34,39,42,43,44,47,48,49
Indriati	10,11,12,14,15,16,17,19,20,21,22,24,25
Lutfi	12,17,27,28,29,32,33,34,37,38,39,41,42,43,44,46,47,48,49
Candra Dewi	4,8,9,11,12,13,14,16,17,21,22,38,43

Tabel 4.12 Contoh Pengecekan *Constraint* 1 pada Individu 1

Bimbingan	Jam/Hari	Dosen 1	Hasil cek *constraint*
28	Rabu, 12.50	Tanzil Furqon	bentrok
37	Selasa, 14.30	Wayan Firdaus	-
23	Rabu, 11.10	Adam Hendra	-
42	Selasa, 15.20	Tanzil Furqon	-
33	Rabu 13.40	Wayan Firdaus	-

Constraint 1 mengharuskan setiap pemilihan jadwal bimbingan terhadap dosen pertama tidak boleh bentrok dengan jadwal mengajar dosen tersebut. jika terjadi bentrok maka akan diberikan bobot pinalti sebesar 1.

Tabel 4.13 Contoh Pengecekan *Constraint* 2 pada Individu 1

Bimbingan	jam hari	Dosen 2	Hasil cek *constraint*
28	Rabu, 12.50	Bayu Rahayudi	-
37	Selasa, 14.30	Indriati	-
23	Rabu, 11.10	Lutfi	Bentrok
42	Selasa, 15.20	Bayu Rahayudi	Bentrok
33	Rabu 13.40	Candra Dewi	-

*Constraint*2 mengharuskan setiap pemilihan jadwal bimbingan terhadap dosen kedua tidak boleh bentrok dengan jadwal mengajar dosen tersebut. jika terjadi bentrok maka akan diberikan bobot pinalti sebesar 1.

Tabel 4.14 Contoh Pengecekan *Constraint* 3 pada Individu 1

Bimbingan	Jam/Hari	Nama Mahasiswa	Jadwal Kuiah	Hasil cek *constraint*
28	Rabu, 12.50	Rizka Rizkiana	37,38,40,42,43,45,47	-
37	Selasa, 14.30	Ana binti	38,43,48	-
23	Rabu, 11.10	Diah Shinta	37,42,47	-
42	Selasa, 15.20	Weni Agustina	4,5,37,38,40,42,43,45,47	Bentrok
28	Rabu, 12.50	Fitri A	-	-

*Constraint*3 mengharuskan setiap pemilihan jadwal bimbingan terhadap mahasiswa tidak boleh ada yang bentrok dengan jadwal perkuliahan mahasiswa. jika terjadi bentrok maka akan diberikan bobot pinalti sebesar 1. Misalkan saja pada kolom yang diberi warna kuning, pada kode 42 terdapat kesamaan pada kode jadwal perkuliahan mahasiswi yang bernama Weni Agustina, maka otomatis akan bentrok dan terdapat pelanggaran maka nilai bobot akan bertambah.

Tabel 4.15 Contoh Pengecekan *Constraint* 4 pada Individu 1

Bimbingan	Jam/Hari	Kode D1	Kode D2	Hasil Cek *Constraint4*
28	Rabu, 12.50	1	2	-
37	Selasa, 14.30	3	4	-
23	Rabu, 11.10	6	7	-
42	Selasa, 15.20	1	2	-
28	Rabu, 12.50	3	10	-

*Constraint*4 mengharuskan setiap pemilihan jadwal bimbingan terhadap mahasiswa yang memiliki dosen pembimbing yang sama tidak boleh bentrok dengan jadwal mahasiswa lainnya. Jika terjadi bentrok maka akan diberikan bobot pinalti sebesar 0,5. Berikut adalah contoh perhitungan *fitness* pada individu 1:

$$Fitness = \frac{k1}{a1+1} + \frac{k2}{a2+1} + \frac{k3}{a3+1} + \frac{k4}{b4+1}$$

$$Fitness_{\text{individu 1}} = \frac{1}{1+1} + \frac{1}{2+1} + \frac{1}{1+1} + \frac{0,5}{0+1} = 1,83$$

Tabel 4.16 Nilai *Fitness* Seluruh Individu

Individu	Nilai
Individu 1	1,833
Individu 2	2,166
Individu 3	2,083

Child 1	2,333
Child 2	2,1667
Child 3	2,75
Child 4	1,75
Child 5	2,333

4.3.4 Proses Seleksi

Proses seleksi ini menggunakan metode *elitism* yang mana menggabungkan populasi awal dan offspring. Setelah digabungkan kemudian akan diurutkan berdasarkan nilai *fitness* yang tertinggi hingga yang terkecil. Lalu akan dipilih sebanyak jumlah *popsize* yang nantinya akan diproses kegenerasi selanjutnya.

Tabel 4.17 Pengurutan Nilai *Fitness* dari Terbesar ke Terkecil

Individu	Nilai	Ranking
Individu 1	1,833	7
Individu 2	2,166	4
Individu 3	2,083	6
Child 1	2,333	2
Child 2	2,1667	5
Child 3	2,75	2
Child 4	1,75	8
Child 5	2,333	3

4.4 Perancangan Pengujian

Perancangan pengujian ini merupakan pengujian yang ada pada parameter algoritme genetika. Dengan adanya pengujian ini bisa diketahui seberapa besar faktor parameter yang ada di algoritme genetika dapat menghasilkan solusi yang optimal dari penjadwalan bimbingan skripsi. Skenario pengujian dalam penelitian ini yaitu:

1. Pengujian ukuran populasi.
2. Pengujian jumlah generasi.
3. Pengujian nilai *cr* dan *mr*.
4. Pengujian nilai bobot.
5. Pengujian konvergensi.

4.4.1 Pengujian Ukuran Populasi

Pengujian ini dilakukan untuk mengukur dan mengetahui ukuran populasi yang optimal yang dihasilkan oleh sistem. Parameter yang digunakan adalah sebagai berikut:

1. Banyaknya generasi = 100
2. *Cr* = 0,4
3. *Mr* = 0,2

Tabel 4.18 Rancangan Pengujian Ukuran Populasi

Uji Coba Ke-*i*	Nilai *Fitness* Terbaik pada Ukuran Populasi									
	10	20	30	40	50	60	70	80	90	100
1										
2										
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10										
Rata-rata										

4.4.2 Pengujian Jumlah Generasi

Pengujian ini dilakukan untuk mengukur dan mengetahui jumlah generasi yang optimal yang dihasilkan oleh sistem. Parameter yang digunakan adalah sebagai berikut:

1. Banyaknya populasi = Nilai terbaik yang telah dilakukan pada sub bab 4.4.1
2. *Crossover rate* = 0,4
3. *Mutation rate* = 0,2

Tabel 4.19 Rancangan Pengujian Jumlah Generasi

Uji Coba Ke-*i*	Nilai *Fitness* Terbaik pada Ukuran Populasi						
	500	1000	1500	2000	2500	3000	3500
1							
2							
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10							
Rata-rata							

4.4.3 Pengujian Nilai *Cr* dan *Mr*

Pengujian nilai *cr* dan *mr* dilakukan untuk mengetahui seberapa besar pengaruh nilai *cr* dan *mr* untuk menghasilkan solusi yang optimal. Tabel 4.20 menunjukan nilai *cr* dan *mr*, yaitu:

Parameter yang digunakan pada pengujian nilai *cr* dan *mr* yaitu:

1. Jumlah generasi = 500
2. Banyaknya populasi = Nilai terbaik yang didapatkan pada sub bab 4.4.1.

Tabel 4.20 Rancangan Pengujian *Crossover Rate* dan *Mutation Rate*

Kombinasi		Nlai *fitness* terbaik pada ukuran populasi										Rata-rata
Cr	***Mr***	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
0,1	0,9											
0,2	0,8											
0,3	0,7											
0,4	0,6											
0,5	0,5											
0,6	0,4											
0,7	0,3											
0,8	0,2											
0,9	0,1											

4.4.4 Pengujian Nilai Bobot Optimal

Pada perancangan pengujian ini digunakan untuk mengetahui nilai bobot ideal yang dihasilkan oleh nilai *fitness* paling optimal. Proses pengujian yaitu pengujian dengan cara menaikan dan menurunkan nilai bobot dari *hard constraint* dan *soft constraint* secara bersamaan (serentak), dan menaikkan nilai bobot *hard constraint* serta menurunkan nilai bobot dari *soft constraint* (tidak serentak).

4.4.4.1 Pengujian Nilai Bobot Secara Serentak

Proses ini dilakukan dengan menambahkan dan mengurangkan nilai bobot *hard constraint* dan *soft constraint* secara bersamaan sesuai dengan variabel pengubahnya. Variabel *x* yaitu variabel nilai pengubah. Pada Tabel 4.21 adalah tabel rancangan pengujian untuk mengubah nilai bobot serentak. Parameter yang digunakan dalam proses pengujian ini adalah:

1. Jumlah Generasi sebesar 1000
2. Banyaknya Populasi adalah nilai terbaik yang didapatkan pada sub bab 4.4.1.
3. *Cr* dan *mr* yaitu nilai terbaik yang didapatkan pada sub bab 4.4.3.

Jika bobot yang dimiliki *hard consraint* bernilai 1 , *soft constraint* bobot bernilai 0,5 dan nilai *x* (variabel pengubah) maka saat x-0,05 penurunan bobot serentak yaitu:

- Bobot *hard constraint* menjadi 0,95
- Bobot *soft constraint* menjadi 0,45

Tabel 4.21 Perancangan Pengujian Nilai Bobot Serentak

Uji coba ke-	Bobot								
	***x*-0,05**	***x*-0,15**	***x*-0,25**	***x*-0,35**	***x*-0,0; *x*+0,0**	***x*+0,05**	***x*+0,15**	***x*+0,25**	***x*+0,35**
1									
2									
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10									

4.4.4.2 Pengujian Nilai Bobot Secara Tidak Serentak

Proses ini dilakukan dengan mengubah nilai bobot untuk *hard constraint* dan *soft constraint* dimana akan dinaikan nilai bobot *hard constraint* sedangkan untuk nilai bobot *soft constraint* akan diturunkan. Variabel y yaitu variabel pengubah nilai bobot. Setiap bobot *hard* costraint akan ditambahkan dengan nilai y dan untuk *soft constraint* akan diturunkan dari nilai *y*. Tabel perancangan pad apengujian ini ditunjukan pada Tabel 4.22. Parameter-parameter algoritme genetika yang digunakan yaitu:

1. Jumlah generasi adalah 1000
2. Banyaknya populasi yaitu nilai terbaik yang didapatkan pada sub bab 4.4.1.
3. *Cr* dan *mr* adalah nilai terbaik yang didapatkan pada sub bab 4.4.3

Jika bobot yang dimiliki *hard consraint* bernilai 1 , *soft constraint* bobot bernilai 0,5 dan nilai *y* (variabel pengubah) maka saat variabel y=0,05 bobot tidak serentak:

- Bobot *hard constraint* menjadi 1,05
- Bobot *soft constraint* menjadi 0,45

Tabel 4.22 Perancangan Pengujian Nilai Bobot Tidak Serentak

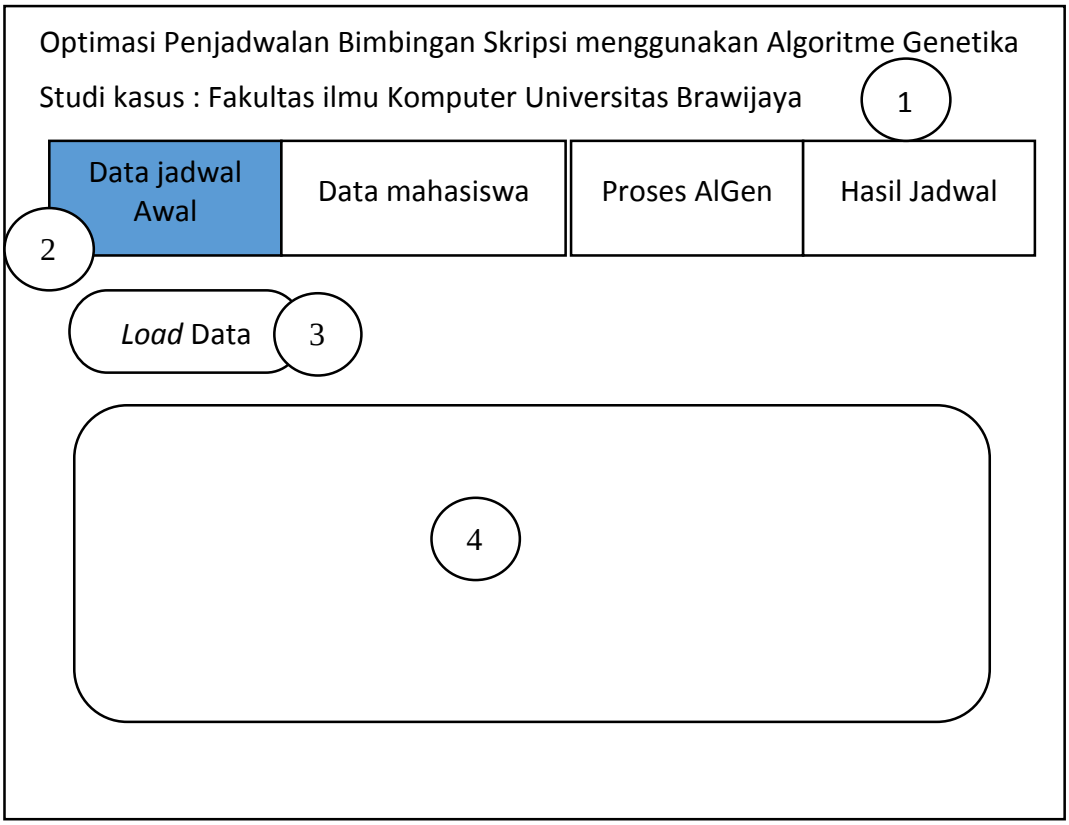
Uji coba ke-*i*	Variabel Pengubahan Nilai Bobot						
	***y*=0,05**	***y*=0, 1**	***y*=0, 15**	***y*=0,2**	***y*=0,25**	***y*=0,3**	***y*=0,35**
1							
2							
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10							

4.5 Perancangan Antarmuka

Perancangan ini bertujuan untuk merancang antarmuka yang akan diimplementasikan oleh penelitian. Pada proses ini terdapat beberapa halaman diantaranya yaitu halaman data jadwal, data mahasiswa, halaman Proses AlGen dan hasil jadwal.

4.5.1 Perancangan Halaman Data Jadwal Awal

Halaman ini adalah tampilan untuk mengakses data jadwal untuk data meliputi *id*, nama mahasiswa, nama dosen pembimbing 1, nama dosen pembimbing 2.



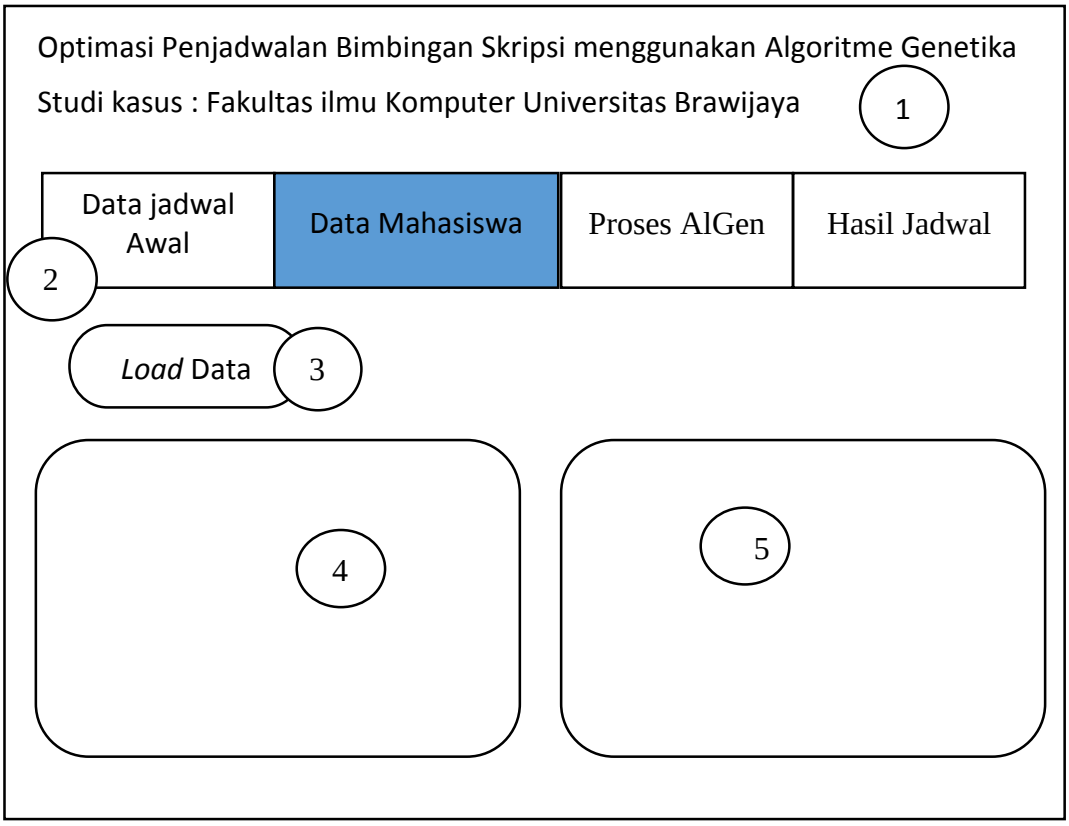
Gambar 4.13 Perancangan Data Jadwal Awal

Keterangan rancangan antarmuka data jadwal awal pada Gambar 4.13 diantaranya yaitu:

1. Judul dari penelitian ini.
2. *Tabbed pane* menu aplikasi terdiri dari Data jadwal awal, data mahasiswa, proses AlGen dan hasil jadwal P0.
3. *Button* untuk memuat data jadwal bimbingan dari *database*
4. Tabel untuk menampilkan data jadwal bimbingan skripsi yang berhasil di *load*.

4.5.2 Perancangan Halaman Data Mahasiswa

Halaman ini yaitu tampilan untuk memuat data mahasiswa yang diakses dari *database*. Data mahasiswa terdiri dari id, nama, dan jadwal kuliah pada semester 8. Sedangkan jadwal dosen terdiri dari *id*, nama dosen dan jadwal mengajar. Pada Gambar 4.14 merupakan perancangan halaman data mahasiswa.



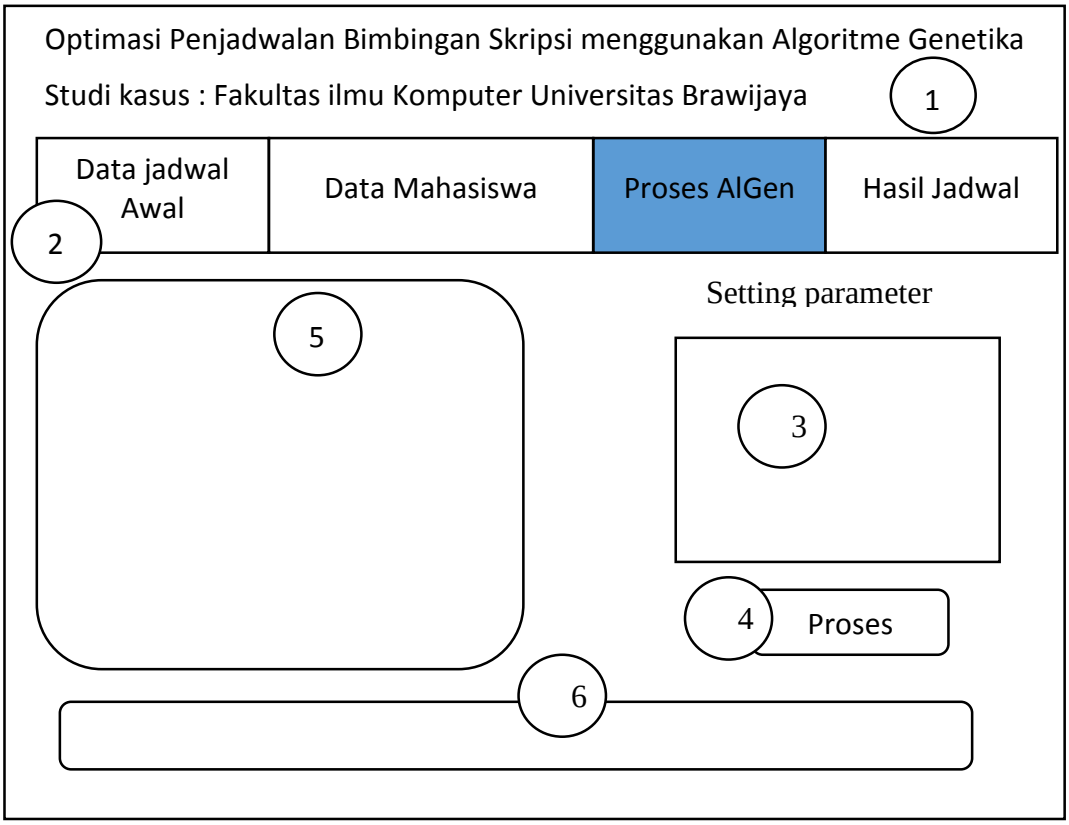
Gambar 4.14 Perancangan Data Mahasiswa

Keterangan rancangan antarmuka data mahasiswa pada Gambar 4.14 diantaranya yaitu:

1. Judul dari penelitian ini.
2. *Tabbed pane* menu aplikasi terdiri dari Data jadwal awal, data mahasiswa, proses AlGen dan hasil jadwal P0.
3. *Button* untuk memuat data mahasiswa dan dosen dari *database*.
4. Tabel untuk menampilkan data dosen yang berhasil di *load*.
5. Tabel untuk menampilkan data mahasiswa yang berhasil di *load*.

4.5.3 Perancangan Halaman Proses Algoritme Genetika

Halaman ini yaitu tampilan untuk memuat data dosen yang diakses dari *database*. Data dosen terdiri dari nip, nama, dan jadwal mengajar. Pada Gambar 4.15 merupakan perancangan halaman *load* data mahasiswa.



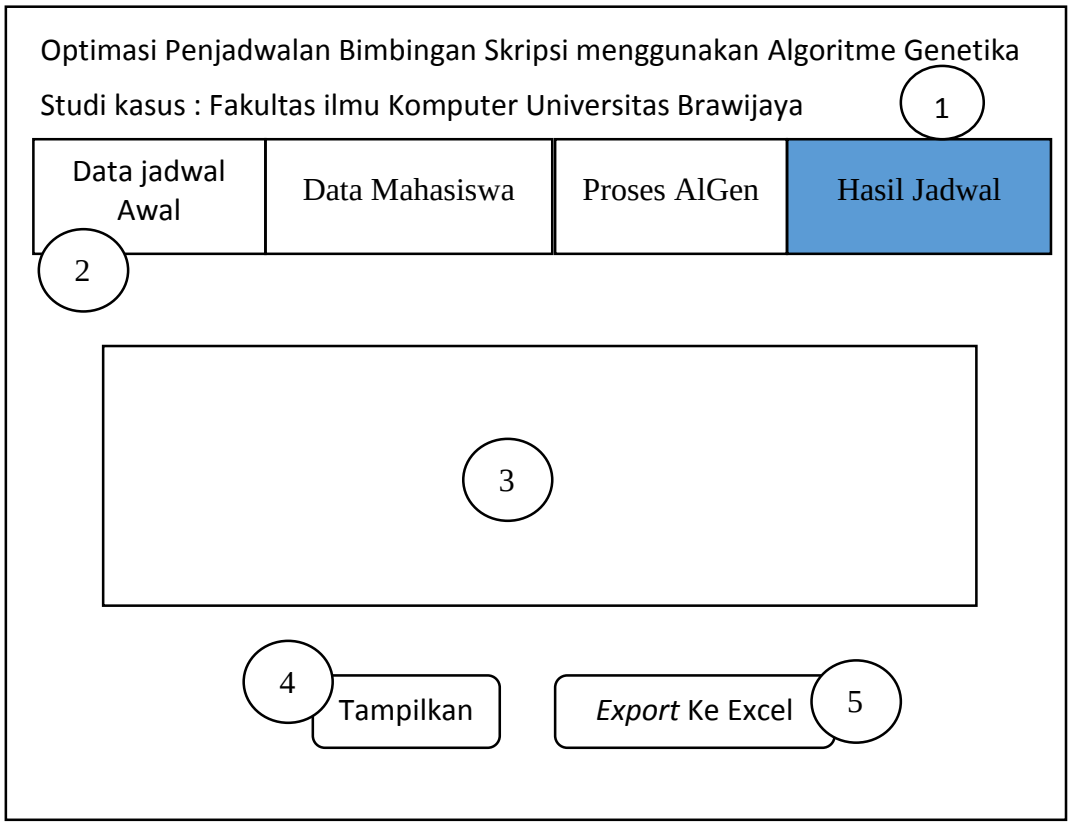
Gambar 4.15 Rancangan Halaman Proses Algen

Keterangan rancangan antarmuka proses algen pada Gambar 4.15 diantaranya yaitu:

1. Judul dari penelitian ini.
2. *Tabbed pane* menu aplikasi terdiri dari Data jadwal awal, data mahasiswa, proses AlGen dan hasil jadwal P0.
3. *Form* untuk *input* nilai dari parameter algoritme genetika seperti *cr*,*mr*,*popsize,* dan generasi.
4. *Botton* untuk proses algoritme genetika.
5. *Table* untuk menampilkan hasil dari proses algoritme genetika yang berisi nilai *fitness* terbaik.
6. *Progress bar* untuk menunjukan bahwa proses perhitungan sedang berjalan.

4.5.4 Perancangan Halaman Proses Hasil Jadwal

Halaman ini adalah tampilan untuk memasukan inialisasi parameter algoritme genetika kedalam sistem. Pengguna memasukan nilai parameter kemudian menekan tombol proses, selanjutnya perhitungan algritma genetika akan di proses dan menampilkannya di Tabel. Perancangan halaman proses algoritme genetika ditunjukan pada Gambar 4.16.



Gambar 4.16 Rancangan Halaman Hasil Jadwal

Keterangan rancangan antarmuka hasil jadwal P0 pada Gambar 4.16 diantaranya yaitu:

1. Judul dari penelitian ini.
2. *Tabbed pane* menu aplikasi terdiri dari Data jadwal awal, data mahasiswa, proses AlGen dan hasil jadwal P0.
3. *Table* untuk menampilkan jadwal bimbngan.
4. *Button* untuk menampilkan individu terbaik hasil perhitungan algoritme genetika menjadi jadwal bimbingan skripsi.
5. *Button* untuk mengeksport hasil jadwal bimbingan kedalam format (.xls).