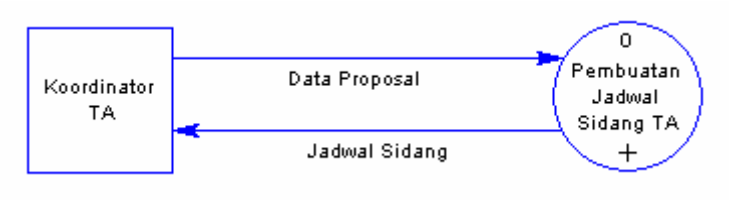
3. PERENCANAAN SISTEM

Pada bab ini, dijelaskan perencanaan sistem berupa analisis terhadap sistem yang ada dan desain sistem baru dengan mengacu pada sistem yang ada tersebut. Secara garis besar perencanaan sistem dapat dijabarkan sebagai berikut:

a. Penggambaran aliran data (*data flow*) dalam sistem.
b. Perencanaan penggunaan *Fuzzy Relation* dalam menentukan dosen penguji.
c. Perencanaan optimasi dengan *Genetic Algorithms*.
d. Perencanaan *database* sistem.

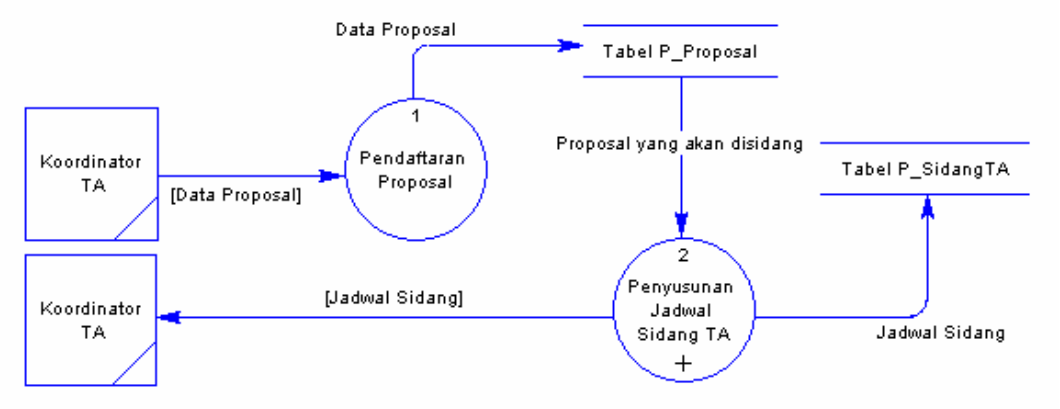
3.1. Aliran Data (*Data Flow*) dalam Sistem

Secara umum, aliran data dalam sistem ini melibatkan Koordinator Tugas Akhir sebagai *user* dan sebuah proses pembuatan jadwal sidang tugas akhir. Aliran data tersebut dapat dilihat pada Gambar 3.1.



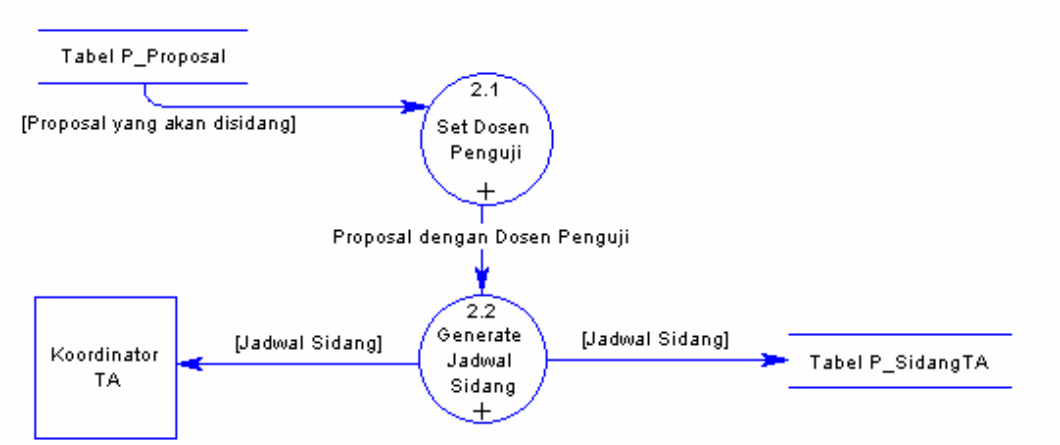
Gambar 3.1. *Context Diagram*

Proses pembuatan jadwal sidang secara garis besar dibagi menjadi dua, yaitu proses pendaftaran proposal dan proses pembuatan jadwal sidang tugas akhir. Proses-proses yang terjadi dalam pembuatan jadwal sidang digambarkan pada Gambar 3.2.



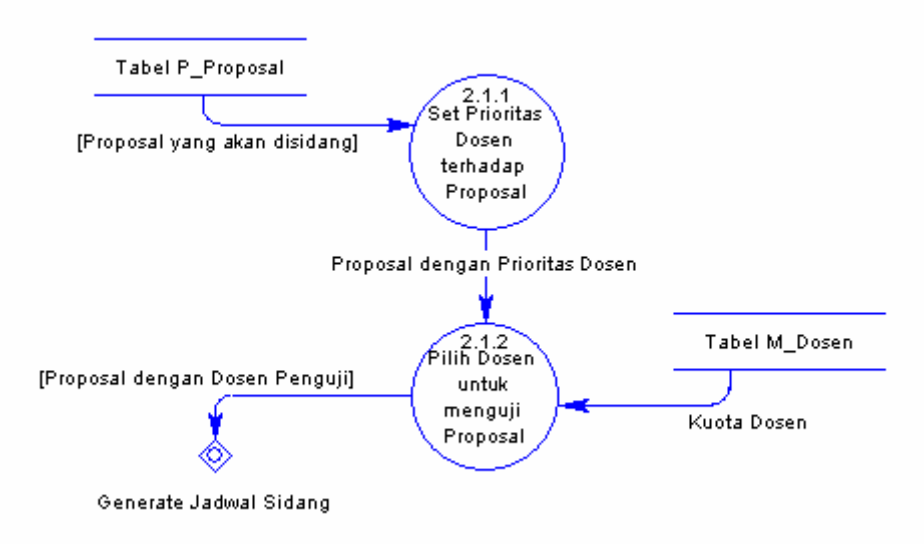
Gambar 3.2. *DFD Level* 0 Pembuatan Jadwal Sidang TA

Proses pendaftaran proposal merupakan proses dimana Koordinator Tugas Akhir menyimpan data proposal mahasiswa ke dalam Tabel P_Proposal. Sedangkan proses penyusunan jadwal sidang tugas akhir terbagi lagi menjadi proses penyusunan dosen penguji dan proses *generate* jadwal sidang, seperti yang digambarkan pada Gambar 3.3.



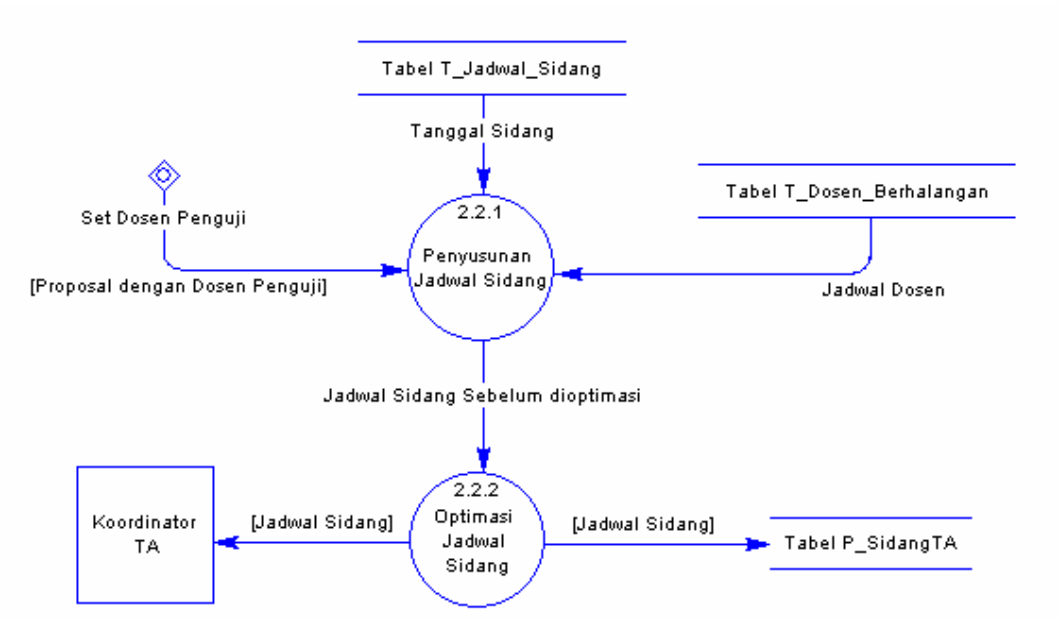
Gambar 3.3. *DFD Level* 1 Penyusunan Jadwal Sidang TA

Set dosen penguji adalah proses dimana dosen penguji diatur untuk menguji suatu tugas akhir dengan menggunakan *Fuzzy Relation*. Sedangkan *generate* jadwal sidang adalah proses men-*generate* jadwal sidang. Kedua proses dalam tersebut dapat dijabarkan lagi seperti yang digambarkan pada Gambar 3.4 dan Gambar 3.5.



Gambar 3.4. *DFD Level* 2 Set Dosen Penguji

Set prioritas dosen adalah proses penyusunan prioritas mahasiswa dari seorang dosen. *Fuzzy Relation* diimplementasikan pada bagian ini.



Gambar 3.5. *DFD Level* 2 Generate Jadwal Sidang

Proses penyusunan dan optimasi jadwal sidang merupakan proses akhir dalam pembuatan jadwal sidang. Optimasi dilakukan dengan menggunakan *Genetic Algorithms*.

Jadwal yang dihasilkan dari sistem ini selain diberikan ke Koordinator Tugas Akhir, juga disimpan dalam Tabel P_Sidang_TA.

3.2. Perencanaan Implementasi *Fuzzy Relation*

Dalam pengembangan *software* penjadwalan sidang tugas akhir ini, *Fuzzy Relation* digunakan untuk menentukan dan menyusun dosen-dosen penguji yang ditugaskan menguji suatu tugas akhir. Penyusunan ini dilakukan dengan harapan agar tugas akhir diuji oleh dosen-dosen yang memiliki keahlian (*expertise*) yang sesuai dengan topik tugas akhir tersebut.

Relasi antara dosen dengan tugas akhir ditentukan dengan melihat relasi antara dosen dengan topik-topik tugas akhir dan tugas akhir dengan topik-topik tugas akhir. Ada tiga hal yang harus diperhatikan yaitu dosen, tugas akhir, dan topik tugas akhir (agar tidak membingungkan, pada uraian berikutnya digunakan kata "topik"). Topik didefinisikan sebagai *domain* $T = \{t_1, t_2, ..., t_n\}$ dimana dosen dan tugas akhir merupakan *fuzzy set* yang berada dalam *domain* T dan didefinisikan sebagai $D = \{D_1, D_2, ..., D_p\}$ dan $P = \{P_1, P_2, ..., P_q\}$, dengan n adalah jumlah topik, p adalah jumlah dosen yang menguji, dan q adalah jumlah tugas akhir yang disidang.

3.2.1. Relasi antara Dosen dan Topik (Nilai *Expertise*)

Dalam relasi ini, dosen D merupakan *fuzzy set* dari topik T yang berarti setiap dosen memiliki nilai *expertise* (keahlian) terhadap setiap topik yang ada. Untuk lebih jelas, dapat dilihat pada contoh berikut.

Misalkan nilai p = 6 dan n = 8. Ini berarti ada 6 dosen dan 8 topik, yang dapat direpresentasikan dalam bentuk

$$D = \{D_1, D_2, D_3, D_4, D_5, D_6\},$$
$$T = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8\},$$

dan kemudian diberikan nilai *fuzzy* sebagai berikut,

$$\mu_{D_1} = \{0.3/t_2, 0.7/t_5, 1/t_7, 1/t_8\},$$
$$\mu_{D_2} = \{1/t_2, 0.8/t_5, 0.8/t_7, 1/t_8\},$$
$$\mu_{D_3} = \{0.9/t_1, 0.9/t_3, 1/t_4, 0.8/t_6\},$$
$$\mu_{D_4} = \{1/t_1, 0.5/t_3, 0.8/t_4, 0.8/t_6\},$$
$$\mu_{D_5} = \{0.1/t_2, 1/t_4, 0.7/t_5, 1/t_8\},$$
$$\mu_{D_6} = \{0.9/t_2, 0.8/t_4, 1/t_5, 1/t_8\}.$$

Arti dari nilai *fuzzy* di atas, misalnya pada dosen D_1, yaitu dosen D_1 memiliki nilai *expertise* terhadap topik t_2 sebesar 30%, topik t_5 sebesar 70, topik t_7 sebesar 100%, dan topik t_8 sebesar 100%. Topik-topik lain yang tidak ditulis memiliki arti bahwa dosen D_1 tidak memiliki keahlian untuk topik-topik tersebut (nilai *expertise* = 0%). Arti yang sama berlaku juga untuk dosen-dosen yang lain.

Nilai *expertise* ini di-*input*-kan oleh *user* dan disimpan ke dalam tabel P_Expertise dengan nilai Kode_Dosen = D_i, nilai Kode_Topik = t_k, dan nilai Nilai_Expertise = e_{ik}, dimana i = 1, 2,..., p; k = 1, 2,..., n; e = nilai *expertise*.

3.2.2. Relasi antara Tugas Akhir dan Topik (Nilai *Content*)

Relasi ini memiliki arti yang sama dengan relasi antara dosen dan topik. Hanya saja pada relasi ini, yang merupakan *fuzzy set* adalah tugas akhir *P*, dimana arti dari relasi ini adalah setiap tugas akhir memiliki nilai *content* (kandungan) terhadap topik. Contoh berikut, menggunakan *domain* topik *T* yang sama dengan contoh pada subbab 3.2.1.

Misalkan nilai q = 5, sehingga ada 5 tugas akhir yang dapat direpresentasikan sebagai berikut

$$P = \{P_1, P_2, P_3, P_4, P_5\},$$

dan nilai *fuzzy* tugas akhir *P* terhadap topik *T* sebagai berikut:

$$\mu_{P_1} = \{0.8/t_2, 0.7/t_3, 1/t_8\},$$
$$\mu_{P_2} = \{1/t_2, 0.8/t_4, 0.9/t_7\},$$
$$\mu_{P_3} = \{0.9/t_1, 1/t_3\},$$
$$\mu_{P_4} = \{1/t_1, 0.7/t_3, 0.8/t_5, 0.3/t_6\},$$
$$\mu_{P_5} = \{1/t_4\}.$$

Nilai tersebut berarti tugas akhir P_1 relevan atau memiliki nilai *content* terhadap topik t_2 sebesar 80%, topik t_3 sebesar 70%, dan topik t_8 sebesar 100%. Pada contoh di atas juga terdapat tugas akhir P_5 yang hanya memiliki sebuah nilai yaitu

$1/t_4$. Maksudnya adalah tugas akhir P_5 benar-benar hanya relevan dengan topik t_4 dengan nilai *content* sebesar 100%.

Nilai *content* ini juga di-*input*-kan oleh *user* dan disimpan dalam tabel T_Proposal_Content dengan nilai Kode_Topik = t_k, dan nilai Nilai_Content = c_{jk}, dimana j = 1, 2,..., q; k = 1, 2,..., n; c = nilai *content*. Tugas akhir dalam *software* ini ditentukan oleh NRP mahasiswa dan urutan proposal yang pernah diajukannya. Karena itu tugas akhir *P* disimpan ke dalam tabel sebagai *field* NRP dan Proposal_Ke. Untuk lebih jelasnya, dapat dilihat ilustrasi pada Subbab 3.4.3.1 tentang Tabel P_Proposal.

3.2.3. Relasi antara Dosen dan Tugas Akhir (Tingkat Kompetensi)

Relasi ini menggambarkan tingkat kompetensi dari dosen terhadap tugas akhir. Setiap anggota dalam *set* dosen dihitung tingkat kompetensinya terhadap setiap anggota dalam *set* tugas akhir. Untuk menghitung tingkat kompetensi tersebut, digunakan rumus (2.2) yang telah disesuaikan sebagai berikut,

$$R(D_i, P_j) = \frac{|D_i \cap P_j|}{|P_j|} = \frac{\sum_{t \in T} \min\{\mu_{D_i}(t), \mu_{P_j}(t)\}}{\sum_{t \in T} \mu_{P_j}(t)} \tag{3.1}$$

Sebagai contoh, dengan menggunakan *domain T* pada contoh 3.2.1, nilai *expertise*

$$\mu_{D_1} = \{0.3/t_2, 0.7/t_5, 1/t_7, 1/t_8\},$$
$$\mu_{D_2} = \{1/t_2, 0.8/t_5, 0.8/t_7, 1/t_8\},$$
$$\mu_{D_3} = \{0.9/t_1, 0.9/t_3, 1/t_4, 0.8/t_6\},$$
$$\mu_{D_4} = \{1/t_1, 0.5/t_3, 0.8/t_4, 0.8/t_6\},$$
$$\mu_{D_5} = \{0.1/t_2, 1/t_4, 0.7/t_5, 1/t_8\},$$
$$\mu_{D_6} = \{0.9/t_2, 0.8/t_4, 1/t_5, 1/t_8\},$$

dan nilai *content*

$$\mu_{P_1} = \{0.8/t_2, 0.7/t_3, 1/t_8\},$$
$$\mu_{P_2} = \{1/t_2, 0.8/t_4, 0.9/t_7\},$$
$$\mu_{P_3} = \{0.9/t_1, 1/t_3\},$$
$$\mu_{P_4} = \{1/t_1, 0.7/t_3, 0.8/t_5, 0.3/t_6\},$$
$$\mu_{P_5} = \{1/t_4\},$$

maka tingkat kompetensi D_1 terhadap P_1 dapat dihitung sebagai berikut:

$$R(D_1, P_1) = \frac{\sum_{t \in T} \min\{\mu_{D_1}(t), \mu_{P_1}(t)\}}{\sum_{t \in T} \mu_{P_1}(t)} = \frac{\min(0.3,0.8) + \min(1,1)}{0.8 + 0.7 + 1} = \frac{1.3}{2.5}$$
$$= 0.52$$

Dengan menggunakan rumus (3.1), diperoleh tingkat kompetensi semua dosen terhadap tugas akhir seperti yang ditunjukkan pada Tabel 3.1.

Tabel 3.1. Tingkat Kompetensi Dosen terhadap Tugas Akhir

$D \setminus P$	P_1	P_2	P_3	P_4	P_5
D_1	0.52	0.44	0	0.25	0
D_2	0.72	0.67	0	0.29	0
D_3	0.28	0.30	0.95	0.68	1
D_4	0.20	0.30	0.74	0.64	0.80
D_5	0.44	0.33	0	0.25	1
D_6	0.72	0.63	0	0.29	0.80

Nilai tingkat kompetensi pada Tabel 3.1 tersebut disimpan dalam Tabel T_Rel_Dosen_Proposal dan digunakan untuk menyusun prioritas setiap dosen terhadap mahasiswa (tugas akhir).

3.2.4. Pemilihan Dosen Penguji

Pemilihan dosen penguji dilakukan dengan memperhatikan pemerataan dosen dan tingkat kompetensi. Pemerataan dosen dihitung dengan menggunakan rumus

$$PROPORSI = \frac{JUMLAH}{PROP\_KUOTA} \tag{3.2}$$

dimana *PROPORSI* adalah nilai proporsi jumlah partisipasi dosen terhadap kuota, *JUMLAH* adalah jumlah dosen berpartisipasi dalam sidang tugas akhir (sebagai pembimbing dan penguji) dan *PROP_KUOTA* adalah perbandingan kuota dosen (untuk perbandingan antara kuota dosen tetap dan kuota dosen luar biasa).

Dosen-dosen diurutkan berdasarkan jumlah proporsi yang paling sedikit. Setiap dosen dibuatkan daftar prioritas mahasiswa yang bisa diuji berdasarkan tingkat kompetensi dosen terhadap tugas akhir mahasiswa. Daftar prioritas ini

diberikan status ketua dan anggota untuk menandai apakah seorang dosen bisa menjadi ketua penguji atau anggota penguji untuk seorang mahasiswa.

Dosen dengan proporsi yang lebih sedikit dipilih dan ditugaskan untuk menguji mahasiswa yang tingkat kompetensinya paling tinggi (mahasiswa yang diprioritaskan). Bila mahasiswa yang tingkat kompetensinya paling tinggi ada lebih dari satu, maka pemilihan dilakukan secara acak dari mahasiswa-mahasiswa tersebut. Sebagai contoh, misalkan dosen D_4 memiliki proporsi yang paling rendah, maka dipilih mahasiswa dengan tugas akhir P_5 yang tingkat kompetensinya paling tinggi untuk dosen D_4, yaitu 0.80.

Hasil dari proses pemilihan ini adalah susunan mahasiswa lengkap dengan dosen pembimbing dan dosen penguji. Hasil ini disimpan pada tabel T_Maju_Sidang untuk persiapan melakukan proses *Genetic Algorithms*.

3.3. Perencanaan Implementasi *Genetic Algorithms*

Dalam merencanakan implementasi *Genetic Algorithms* untuk *software* optimasi penjadwalan sidang tugas akhir, dibutuhkan lima langkah penting, yaitu pengkodean kromosom, pembuatan populasi awal, proses seleksi, proses regenerasi (*crossover*, mutasi, dan reproduksi), dan pengulangan proses seleksi dan regenerasi.

3.3.1. Pengkodean Kromosom (*Encoding)*

Pengkodean kromosom dalam *software* ini adalah bagaimana cara suatu susunan jadwal sidang tugas akhir dituliskan dalam bentuk susunan gen dalam kromosom. Susunan gen inilah yang nantinya mengalami proses *crossover*, mutasi, dan regenerasi.

Susunan jadwal sidang tugas akhir dibagi berdasarkan tanggal, sesi, dan ruang sidang seperti yang dapat dilihat pada Tabel 3.2.

Tabel 3.2. Susunan Jadwal Sidang

	Sesi 1		Sesi 2		Sesi 3	
Tanggal 1	Ruang 1	Ruang 2	Ruang 1	Ruang 2	Ruang 1	Ruang 2
Tanggal 2	Ruang 1	Ruang 2	Ruang 1	Ruang 2	Ruang 1	Ruang 2
Tanggal 3	Ruang 1	Ruang 2	Ruang 1	Ruang 2	Ruang 1	Ruang 2
Tanggal 4	Ruang 1	Ruang 2	Ruang 1	Ruang 2	Ruang 1	Ruang 2

Mengacu pada Tabel 3.2, susunan gen disusun dalam bentuk matriks tiga dimensi. Susunan gen didefinisikan sebagai berikut:

```
Gen[0..T-1, 0..S-1, 0..R-1]
```

dimana *T* menyatakan jumlah tanggal, *S* menyatakan jumlah sesi, dan *R* menyatakan jumlah ruang. Bentuk matriks pengkodean gen dari Tabel 3.2 adalah sebagai berikut:

```
[0,0,0]  [0,0,1]  [0,1,0]  [0,1,1]  [0,2,0]  [0,2,1]
[1,0,0]  [1,0,1]  [1,1,0]  [1,1,1]  [1,2,0]  [1,2,1]
[2,0,0]  [2,0,1]  [2,1,0]  [2,1,1]  [2,2,0]  [2,2,1]
[3,0,0]  [3,0,1]  [3,1,0]  [3,1,1]  [3,2,0]  [3,2,1]
```

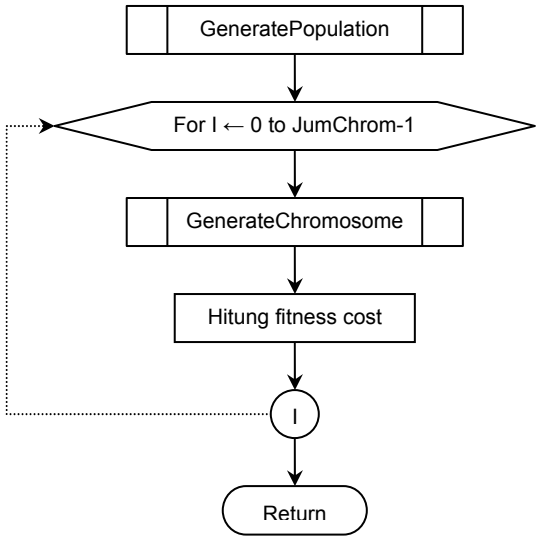
Setiap gen pada matriks tersebut memiliki nilai (*allele*) berupa NRP, Pembimbing 1, Pembimbing 2, Ketua Penguji, Penguji 1, dan Penguji 2 yang diperoleh dari hasil implementasi *Fuzzy Relation*. Selain itu, setiap gen juga menyimpan *fitness cost* terhadap jadwal dan *fitness cost* terhadap tingkat kompetensi (perhitungan *fitness cost* ini dijelaskan pada Subbab 3.3.2.2 tentang *Fitness Cost*). Jadi setiap gen memiliki delapan nilai, yaitu NRP mahasiswa yang disidang, Dosen Pembimbing 1, Dosen Pembimbing 2, Dosen Ketua Penguji, Dosen Penguji 1, Dosen Penguji 2, *fitness cost* jadwal, dan *fitness cost* kompetensi. Indeks dari gen tersebut menyatakan jadwal sidang (tanggal, sesi, dan ruang sidang) sehingga gen ini dinamakan ***Gen Jadwal***.

Dengan pengkodean ini, diperoleh sebuah kromosom (kandidat solusi) dengan gen-gen yang memiliki nilai berupa NRP mahasiswa yang disidang dan dosen-dosen, serta indeks untuk mewakili tanggal, sesi, dan ruang sidang.

Kromosom inilah yang dipakai dalam seluruh proses *Genetic Algorithms* pada *software* ini.

3.3.2. Pembuatan Populasi Awal

Setelah melakukan pengkodean, langkah berikutnya dalam *Genetic Algorithms* adalah membentuk populasi awal yang terdiri dari beberapa kromosom. Langkah-langkah dalam membuat populasi awal adalah pembuatan kromosom dan perhitungan *fitness cost*. Gambar 3.6 menunjukkan *flowchart* dalam pembuatan populasi awal.



Gambar 3.6. *Flowchart* Pembuatan Populasi Awal

3.3.2.1. Pembuatan Kromosom

Dalam pembentukan kromosom, ada dua hal yang harus dihindari agar kromosom tersebut memiliki *fitness cost* yang bagus, yaitu:

a. Dosen bentrok terhadap jadwal yang sama. Hal ini terjadi bila pada tanggal dan sesi yang sama, seorang dosen ditugaskan pada dua atau lebih ruang paralel. Kondisi ini tidak bisa ditoleransi sehingga kromosom yang memiliki kondisi seperti ini dianggap tidak valid (semua *fitness cost* diberi nilai 0).
b. Dosen dijadwalkan pada tanggal dan sesi dimana dosen tersebut berhalangan. Kondisi ini masih dapat ditoleransi, namun berakibat pada berkurangnya *fitness cost*.

Dalam pengembangan *software* ini, jumlah kromosom yang ada dalam satu populasi sebanyak 4 buah. Keempat kromosom ini memiliki indeks dari 0 sampai 3 dan cara penyusunan setiap kromosom berbeda-beda tergantung indeks. Cara penyusunan kromosom-kromosom tersebut adalah:

a. Kromosom dengan indeks 0 disusun berdasarkan NRP *ascending* dengan prioritas dosen luar biasa diutamakan.
b. Kromosom dengan indeks 1 disusun berdasarkan NRP *descending* dengan prioritas dosen luar biasa diutamakan.
c. Kromosom dengan indeks 2 disusun berdasarkan NRP *ascending* dengan prioritas dosen yang memiliki jumlah jadwal bisa lebih sedikit diutamakan.
d. Kromosom dengan indeks 3 disusun berdasarkan NRP *descending* dengan prioritas dosen yang memiliki jumlah jadwal bisa lebih sedikit diutamakan.

Pengisian seperti ini bertujuan untuk menghasilkan susunan gen yang berbeda antara kromosom yang satu dengan yang lain.

Untuk mendapatkan kromosom dengan *fitness cost* kompetensi yang bagus, pengisian dilakukan dengan memperhatikan susunan kelima dosen. Prioritas yang digunakan dalam melakukan pengisian gen adalah:

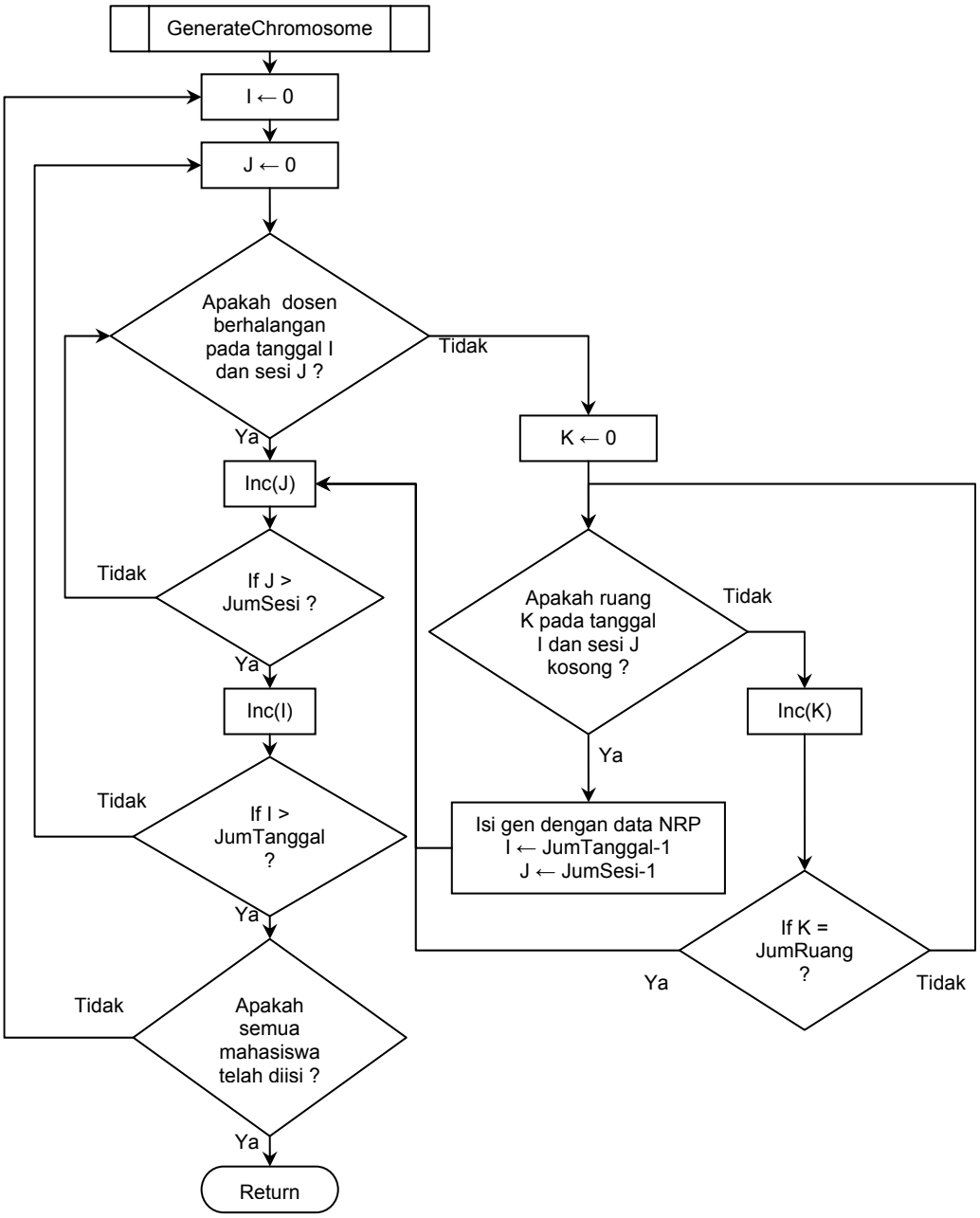
a. Kelima dosen bisa hadir dalam tanggal dan sesi yang dijadwalkan.
b. Salah satu anggota penguji tidak bisa hadir. Hanya ada pembimbing, ketua penguji, dan satu anggota penguji.
c. Kedua anggota penguji tidak bisa hadir. Hanya ada pembimbing dan ketua penguji.
d. Ketua penguji tidak bisa hadir. Hanya ada pembimbing dan kedua anggota penguji.
e. Ketua penguji dan salah satu anggota penguji tidak bisa hadir. Hanya ada pembimbing dan satu anggota penguji.
f. Semua penguji tidak bisa hadir.
g. Semua dosen tidak bisa hadir.

Prioritas ini mempengaruhi *fitness cost* jadwal dimana semakin ke bawah nilai *fitness cost* jadwal menjadi semakin rendah.

Pada awal penyusunan, semua gen memiliki nilai NRP 0 yang berarti jadwal sidang yang diwakili gen tersebut masih kosong. Pada akhir penyusunan,

jumlah gen yang terisi dengan NRP mahasiswa sebanyak jumlah mahasiswa yang disidang. Gen-gen lain yang tidak terisi tetap dipertahankan dengan alasan pada saat *crossover* dan mutasi, nilai dari gen-gen yang telah terisi dapat berpindah ke gen-gen yang masih kosong, yang berarti mahasiswa tersebut mengalami pergantian jadwal sidang.

Gambar 3.7 menunjukkan *flowchart* untuk membuat kromosom awal.



Gambar 3.7. *Flowchart Generate* Kromosom

3.3.2.2. *Fitness Cost*

Sebelum menghitung *fitness cost*, perlu dilakukan penyusunan prioritas, dimana prioritas yang lebih tinggi lebih diutamakan dalam proses seleksi. Urutan prioritas dalam *software* ini dapat dijelaskan sebagai berikut:

a. Prioritas pertama yaitu jadwal berhalangan seorang dosen terhadap jadwal sidang. Jadwal sidang yang diberikan kepada seorang dosen diharapkan tidak bertepatan dengan jadwal berhalangan dosen tersebut. Terutama untuk dosen pembimbing dan ketua penguji. Bila seorang dosen tidak berhalangan maka *fitness cost* ditambahkan dengan sebuah nilai tertentu yang dinamakan ***Koefisien Jadwal***. Nilai tersebut dapat dilihat pada Tabel 3.3.

Tabel 3.3. Koefisien Jadwal untuk Dosen

Dosen Tidak Berhalangan	**Koefisien Jadwal (J)**
Pembimbing 1 (B_1)	6
Pembimbing 2 (B_2)	4
Ketua Penguji (KU)	3
Penguji 1 (U_1)	1
Penguji 2 (U_2)	1

Dosen Penguji 1 dan Penguji 2 diberi Nilai Jadwal yang sama dengan alasan bahwa kedua dosen itu memiliki status yang sama dalam sidang yaitu sebagai anggota penguji.

Dengan menggunakan nilai pada Tabel 3.3, rumus untuk menghitung *fitness cost* jadwal dapat didefinisikan sebagai berikut:

$$FC_{Jadwal} = (J_{B1} \times B_1) + (J_{B2} \times B_2) + (J_{KU} \times KU) + (J_{U1} \times U_1) + (J_{U2} \times U_2) \quad (3.3)$$

dimana B_1, B_2, KU, U_1, dan U_2 bernilai 1 jika dosen bisa dan bernilai 0 jika dosen berhalangan.

Fitness cost yang ditambahkan adalah *fitness cost* dari gen dimana dosen tersebut berada. Sebagai contoh, misalkan sebuah gen G memiliki dosen B_1, B_2, KU, U_1, dan U_2, dengan kondisi dosen U_1 berhalangan hadir dalam sidang. Maka *fitness cost* dari gen tersebut dapat dihitung sebagai berikut:

$$FC_{Jadwal} = (6 \times 1) + (4 \times 1) + (3 \times 1) + (1 \times 0) + (1 \times 1) = 14$$

Fitness cost ini memiliki nilai maksimal 15 yang berarti semua dosen bisa hadir dalam sidang.

Untuk menghitung *fitness cost* dari kromosom, diambil nilai minimum *fitness cost* jadwal dari semua gen yang ada mahasiswanya (gen yang terisi).

b. Prioritas kedua adalah penentuan dosen yang menguji sebuah tugas akhir. Dosen yang menguji sebuah tugas akhir diharapkan mengerti dan memahami topik dari tugas akhir tersebut. Nilai yang digunakan untuk menghitung *fitness cost* ini adalah Nilai_Relasi dari Tabel T_Rel_Dosen_Proposal yang merupakan tingkat kompetensi dari dosen.

   Ketua penguji mendapatkan bobot (*K*) yang lebih tinggi daripada anggota penguji, sehingga nilai ketua penguji dikalikan 5, sedangkan nilai anggota penguji dikalikan 2. Rumus untuk menghitung *fitness cost* kompetensi dapat didefinisikan sebagai berikut:

$$FC_{Kompetensi} = (K_{KU} \times N_{KU}) + (K_{U1} \times N_{U1}) + (K_{U2} \times N_{U2}) \quad (3.4)$$

   dimana *N* adalah tingkat kompetensi dosen terhadap tugas akhir.

   Sebagai contoh, misalkan sebuah gen memiliki tingkat kompetensi ketua penguji sebesar 0.85, penguji 1 sebesar 0.92, dan penguji 2 sebesar 0.80. *Fitness cost* gen tersebut dapat dihitung sebagai berikut:

$$FC_{Kompetensi} = (5 \times 0.85) + (2 \times 0.92) + (2 \times 0.80) = 7.69 .$$

   Untuk *fitness cost* kromosom, juga diambil nilai minimum *fitness cost* kompetensi dari semua gen yang terisi, sama seperti *fitness cost* jadwal.

Penggunaan *fitness cost* minimum ini bertujuan untuk menghindari pemilihan kromosom dengan *fitness cost* rata-rata (*average*) yang bagus, tetapi memiliki satu gen yang jelek sekali (memiliki *fitness cost* gen yang rendah sekali). Namun, selain menggunakan *fitness cost* minimum, *fitness cost* rata-rata juga digunakan dengan tujuan sebagai penentu bila ternyata ada dua kromosom yang memiliki nilai minimum yang sama.

3.3.3. Proses Seleksi

Setelah memiliki sebuah populasi dengan empat kromosom, langkah berikutnya adalah melakukan seleksi. Tujuan proses seleksi ini adalah untuk

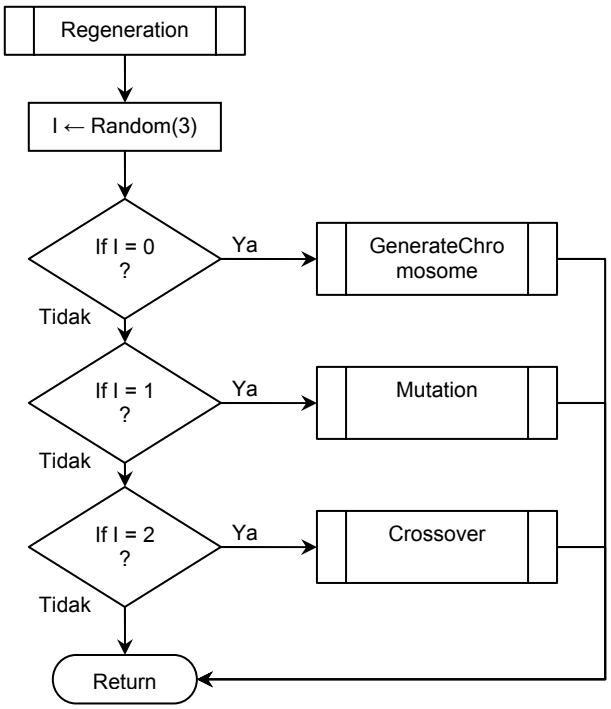
mencari kromosom terbaik (memiliki *fitness cost* yang tinggi) dalam satu generasi. Proses seleksi dilakukan dengan metode *Elitism*, dimana satu kromosom terbaik disimpan untuk diikutsertakan pada proses seleksi generasi berikutnya.

Seleksi dilakukan dengan cara memilih kromosom terbaik dari kromosom-kromosom yang ada.. Untuk generasi pertama, kromosom pertama ini langsung disimpan sebagai kromosom terbaik. Sedangkan untuk generasi berikutnya, kromosom pertama dibandingkan dengan kromosom terbaik. Bila kromosom pertama lebih baik dari kromosom terbaik, maka kromosom terbaik diganti dengan kromosom pertama tersebut.

Proses pembandingan dua kromosom dilakukan dengan cara membandingkan *fitness cost* berdasarkan urutan prioritas pada Subbab 3.2.2.2. Yang pertama dilakukan adalah membandingkan *fitness cost* jadwal kedua kromosom, kemudian membandingkan *fitness cost* jadwal rata-rata, setelah itu *fitness cost* kompetensi, dan yang terakhir *fitness cost* kompetensi rata-rata.

3.3.4. Proses Regenerasi

Proses regenerasi adalah proses pembentukan kromosom-kromosom baru untuk menggantikan generasi lama dengan tujuan mencari kromosom yang memiliki nilai yang lebih bagus. Proses regenerasi dilakukan dengan tiga cara yaitu *crossover*, mutasi, dan reproduksi. Ketiga cara ini tidak dilakukan sekaligus dalam setiap generasi, tetapi dilakukan proses pengacakan untuk memilih cara regenerasi untuk setiap generasi. Perbandingan *crossover*, mutasi, dan reproduksi adalah 1 : 1 : 1. *Flowchart* proses regenerasi dapat dilihat pada Gambar 3.8.



Gambar 3.8. *Flowchart* Proses Regenerasi

3.3.4.1. *Crossover*

Crossover dilakukan dengan memilih dua kromosom secara acak, kedua kromosom tidak boleh sama, dan kemudian menukar sebagian gen dari kedua kromosom itu. Metode yang digunakan adalah *two-point crossover*. Dua buah posisi dipilih secara acak dan semua gen yang berada di antara dua posisi tersebut ditukar. Setelah melakukan penukaran gen, dilakukan penyesuaian gen-gen yang tidak di-*crossover* terhadap gen-gen baru tersebut. Penyesuaian ini dilakukan agar tidak ada nilai NRP yang sama di antara gen dalam satu kromosom. *Crossover* hanya terjadi pada level mahasiswa.

Contoh *crossover*:

Kromosom 1

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1	NRP2	NRP3	NRP4		NRP5	NRP6	NRP7

Kromosom 2

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		NRP5	NRP4	NRP2	NRP6	NRP7	NRP3

Misalkan posisi yang dipilih secara acak adalah gen dengan indeks [0,2,0] dan [1,1,0]. Hasil *crossover* dari kedua kromosom tersebut adalah:

Kromosom 1

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1	NRP2	**NRP5**	**NRP4**	**NRP2**	**NRP6**	NRP6	NRP7

Kromosom 2

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		**NRP3**	**NRP4**		**NRP5**	NRP7	NRP3

Penyesuaian yang terjadi pada kromosom 1 adalah:

a. Gen [0,1,0] sama dengan gen [1,0,0], maka gen [0,1,0] mengambil nilai gen [1,0,0] sebelum di-*crossover*, yaitu nilai kosong.

Perubahan 1

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		**NRP5**	**NRP4**	**NRP2**	**NRP6**	NRP6	NRP7

b. Gen [1,2,0] sama dengan gen [1,1,0], maka gen [1,2,0] mengambil nilai gen [1,1,0] sebelum di-*crossover*, yaitu *NRP5*.

Perubahan 2

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		**NRP5**	**NRP4**	**NRP2**	**NRP6**	NRP5	NRP7

c. Gen [1,2,0] yang mengalami penyesuaian sama dengan gen [0,2,0]. Karena gen [0,2,0] mengalami *crossover*, maka nilai gen [1,2,0] harus mengalami penyesuaian lagi, sehingga menjadi *NRP3*.

Perubahan 3

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		**NRP5**	**NRP4**	**NRP2**	**NRP6**	NRP3	NRP7

Penyesuaian yang terjadi pada kromosom 2 adalah:

a. Gen [1,3,0] sama dengan gen [0,2,0], maka gen [1,3,0] mengambil nilai gen [0,2,0] sebelum di-*crossover*, yaitu *NRP5*.

Perubahan 1

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		**NRP3**	**NRP4**		**NRP5**	NRP7	NRP5

b. Gen [1,3,0] yang mengalami penyesuaian sama dengan gen [1,1,0]. Karena gen [1,1,0] mengalami *crossover*, maka gen [1,3,0] harus mengalami penyesuaian lagi, sehingga menjadi *NRP6*.

Perubahan 2

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		**NRP3**	**NRP4**		**NRP5**	NRP7	NRP6

c. Gen [1,0,0] memiliki nilai kosong. Karena nilai gen sebelum di-*crossover* tidak kosong dan gen tersebut tidak sama dengan gen-gen yang di-*crossover*, maka nilai yang hilang ini harus dicarikan tempat kosong dalam kromosom (Bila nilai gen sebelum di-*crossover* kosong, tidak perlu dilakukan penyesuaian).

Perubahan 3

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1	NRP2	**NRP3**	**NRP4**		**NRP5**	NRP7	NRP3

Jadi setelah mengalami *crossover*, kedua kromosom tersebut menjadi:

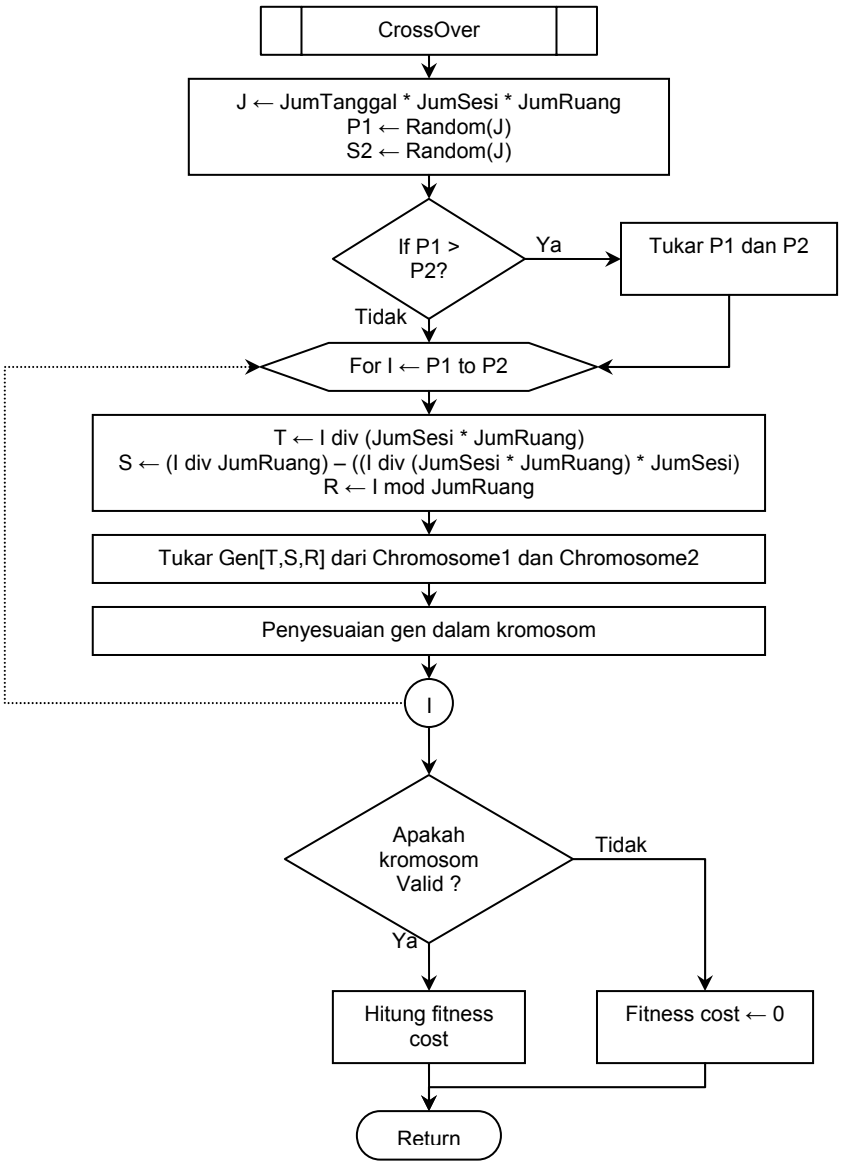
Kromosom 1

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1		**NRP5**	**NRP4**	**NRP2**	**NRP6**	NRP3	NRP7

Kromosom 2

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1	NRP2	**NRP3**	**NRP4**		**NRP5**	NRP7	NRP6

Gambar 3.9 menunjukkan *flowchart crossover*.



Gambar 3.9. *Flowchart Crossover*

3.3.4.2. Mutasi

Mutasi dilakukan dengan cara memilih dua gen secara random dari sebuah kromosom, dan kemudian ditukar nilainya. Mutasi dilakukan terhadap seluruh kromosom yang ada dalam satu generasi. Mutasi dapat terjadi pada dua level yaitu level mahasiswa dan level dosen.

Pada level mahasiswa, mutasi menukar seluruh nilai gen (satu jadwal sidang) dari dua gen yang terpilih. Penukaran ini memungkinkan sebuah gen

berpindah ke gen yang kosong. Dengan alasan inilah maka gen-gen yang kosong tetap dipertahankan.

Contoh mutasi pada level mahasiswa:

Kromosom awal:

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1	NRP2	NRP3	NRP4		NRP5	NRP6	NRP7

Misalkan dua gen yang dipilih secara random adalah gen dengan indeks [0,2,0] dan [1,2,0]. Hasil mutasi dari kromosom tersebut adalah:

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1	NRP2	**NRP6**	NRP4		NRP5	**NRP3**	NRP7

Bila dari kromosom awal, gen yang terpilih adalah gen [0,2,0] dan gen [1,0,0] yang kosong, maka hasil mutasi menjadi:

000	*010*	*020*	*030*	*100*	*110*	*120*	*130*
NRP1	NRP2		NRP4	**NRP3**	NRP5	NRP6	NRP7

Gen yang kosong sekarang berada pada gen [0,2,0].

Pada level dosen, mutasi dilakukan terhadap nilai dosen penguji dalam gen, baik itu ketua penguji, penguji 1, maupun penguji 2. Gen-gen yang ditukar pada level ini harus merupakan gen yang sudah terisi. Hal ini dilakukan untuk menghindari kondisi yang tidak valid dimana dosen penguji ditempatkan pada gen yang tidak memiliki mahasiswa yang disidang.

Contoh mutasi pada level dosen:

Kromosom awal (yang ditulis hanya dosen penguji):

000	*001*	*010*	*011*	*100*	*101*	*110*	*111*
ABC	DEF	ADF	BHG		CEH	BDF	

Misalkan terjadi mutasi pada gen [0,0,1] dosen 1 dan gen [1,0,1] dosen 3, maka kromosom tersebut menjadi:

000	*001*	*010*	*011*	*100*	*101*	*110*	*111*
ABC	**H**EF	ADF	BHG		CE**D**	BDF	

Pada level ini, kondisi dimana terjadi mutasi pada gen [0,0,1] dan [1,0,0] tidak mungkin terjadi karena sudah dilakukan pengecekan terhadap isi gen (lihat Gambar 3.11). Namun, kondisi tidak valid masih dapat terjadi seperti bila mutasi terjadi pada gen [0,0,1] dosen 1 dan gen [1,1,0] dosen 1. Hasil mutasi adalah sebagai berikut:

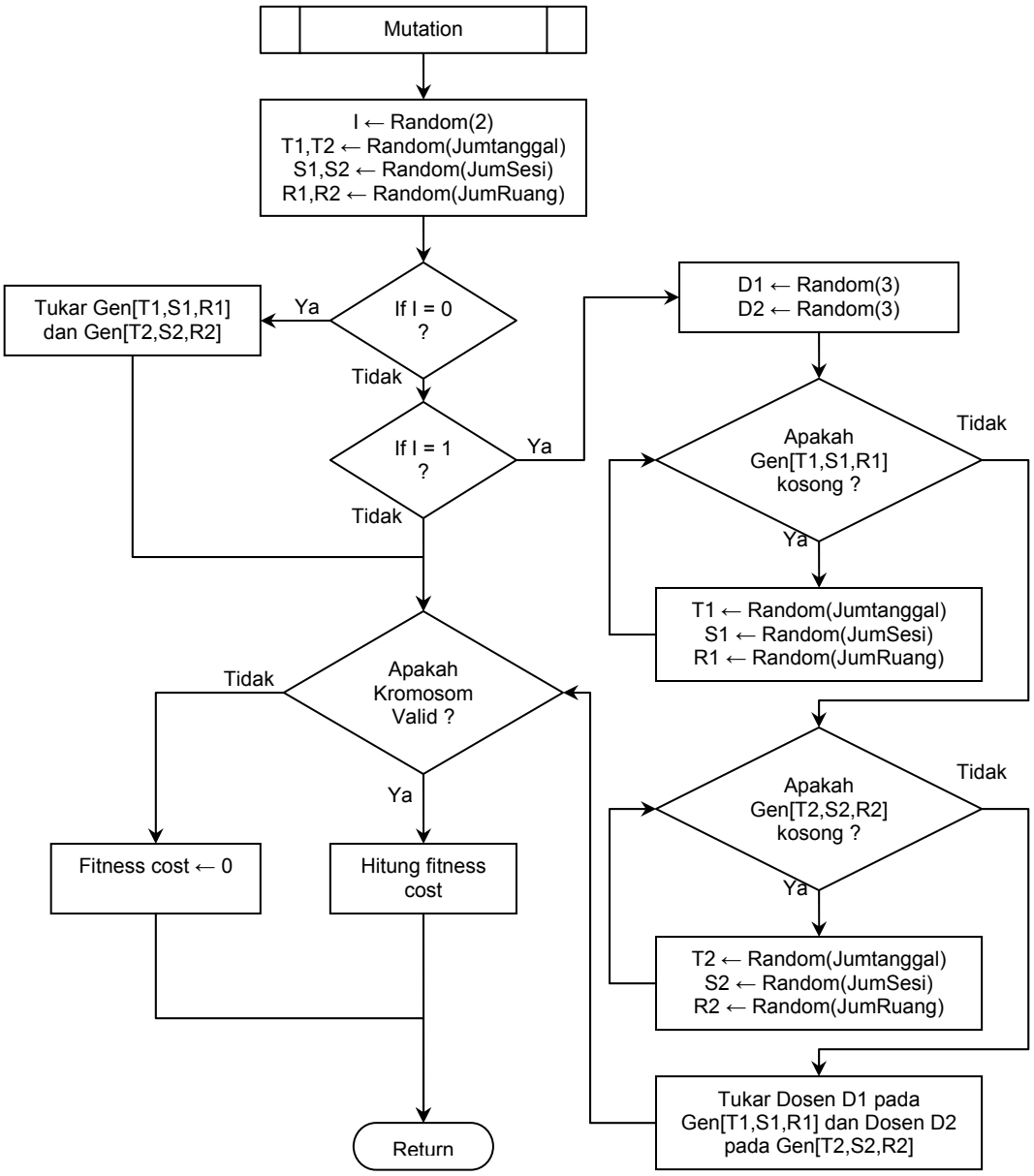
000	*001*	*010*	*011*	*100*	*101*	*110*	*111*
ABC	**B**EF	ADF	BHG		CEH	**D**DF	

Gen [0,0,1] memiliki dosen *B* yang juga berada pada gen [0,0,0]. Indeks pertama dan kedua yang menyatakan tanggal dan sesi sidang sama yang berarti dosen *B* tersebut menguji pada tanggal dan sesi yang sama namun pada ruang yang berbeda. Kondisi ini adalah kondisi yang tidak valid.

Kondisi tidak valid yang lain juga terjadi pada gen [1,1,0] dimana pada gen tersebut Dosen *D* ditulis dua kali yang berarti pada tanggal, sesi, dan ruang yang sama, ada dua orang Dosen *D*.

Kondisi-kondisi tidak valid ini juga dilihat terhadap dosen pembimbing, sehingga tidak ada dosen yang jadwal membimbing dan mengujinya bentrok ataupun berada dalam satu sidang sebagai pembimbing dan penguji.

Gambar 3.10 menunjukkan *flowchart* mutasi.



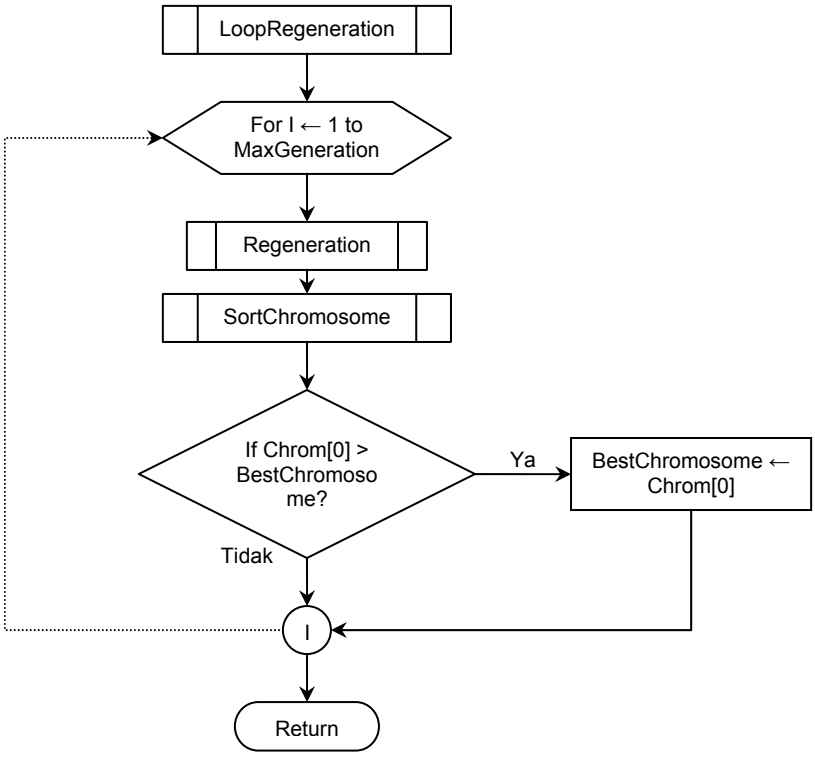
Gambar 3.10. *Flowchart* Mutasi

3.3.4.3. Reproduksi

Reproduksi merupakan pembuatan ulang kromosom yang sama seperti pada pembuatan populasi awal. Tetapi, tidak semua kromosom yang ada pada generasi ini diganti. Kromosom yang diganti dipilih secara acak sebanyak dua buah dan kemudian disusun lagi dengan mengikuti aturan indeks pada Subbab 3.3.2.1.

3.3.5. Pengulangan Proses Seleksi dan Regenerasi

Langkah terakhir adalah melakukan pengulangan proses seleksi dan regenerasi sebanyak *input*-an jumlah generasi dari *user*. Proses pengulangan ini dapat dilihat pada Gambar 3.11.



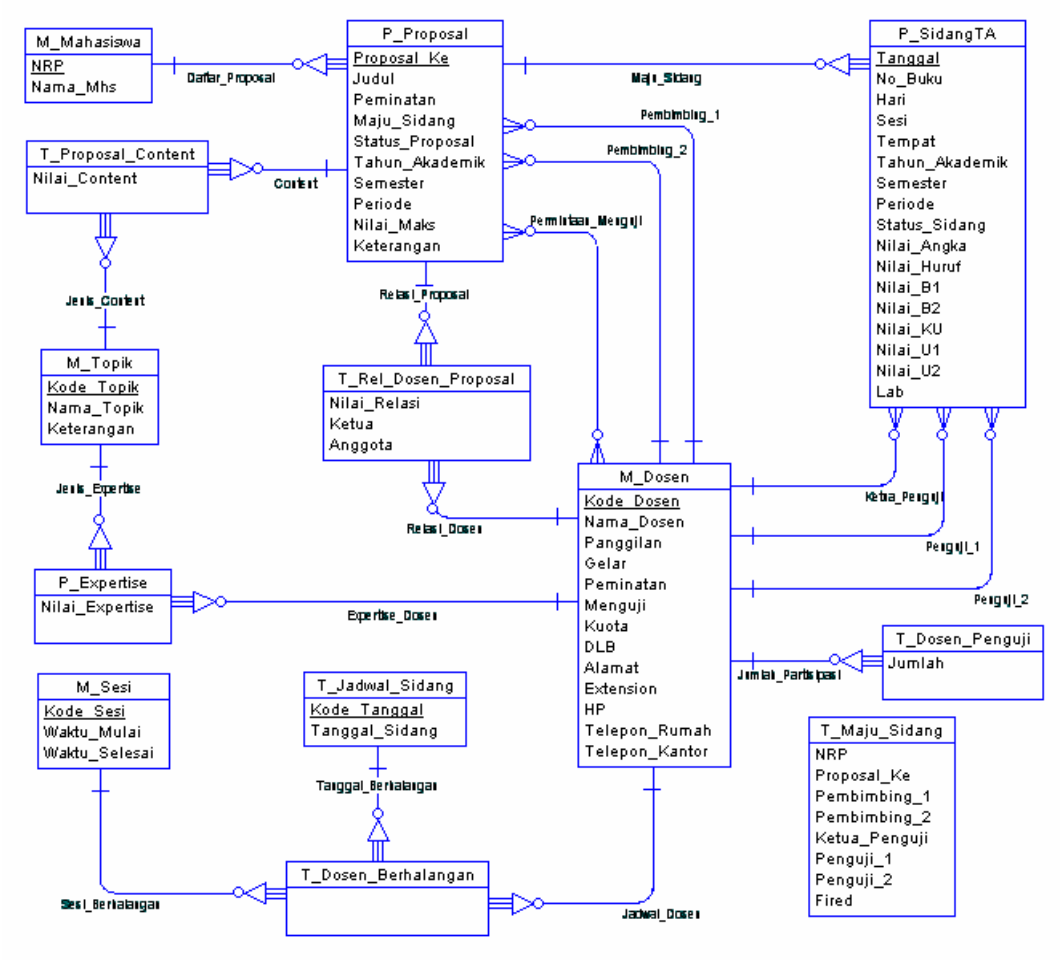
Gambar 3.11. *Flowchart* Pengulangan Proses Seleksi dan Regenerasi

3.4. Perencanaan *Database* Sistem

Perencanaan *database* tetap mengacu pada sistem yang ada sekarang. Tujuan dari pembuatan *database* baru adalah untuk menyimpan data-data yang digunakan dalam *Genetic Algorithms*, yang bersifat *temporary* (sementara). Ini dilakukan untuk memisahkan data-data *temporary* tersebut dengan data-data yang disimpan sebagai informasi (*fixed*) dalam *database* lama.

3.4.1. *Conceptual Model* Sistem

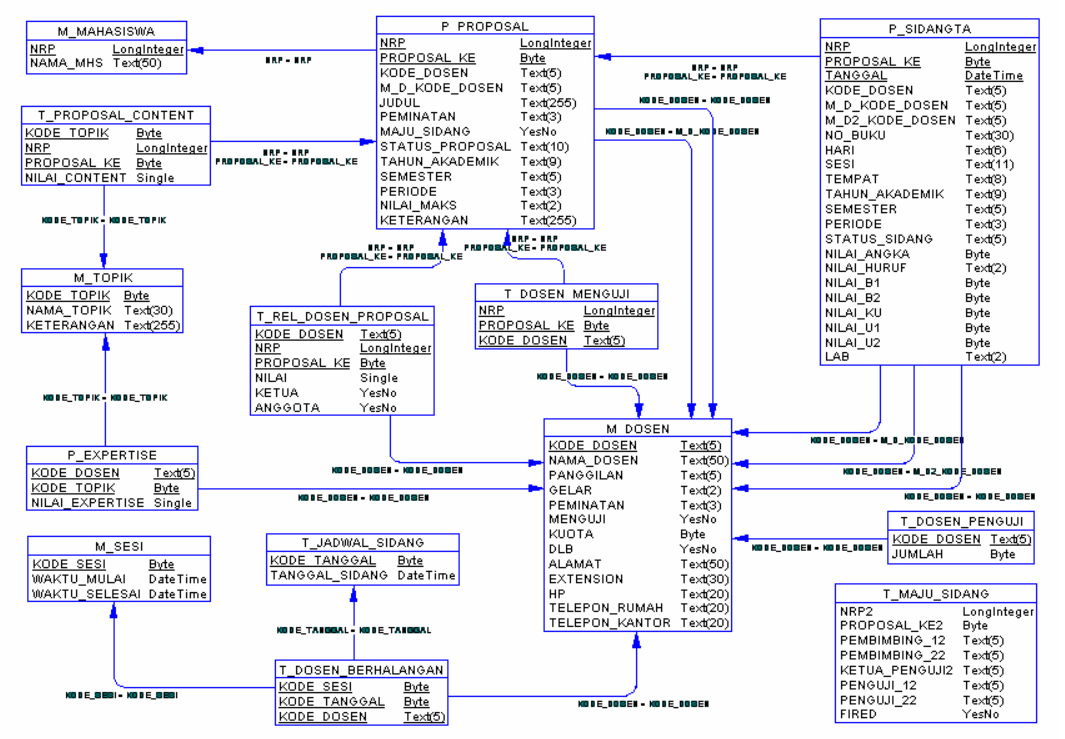
Conceptual Model merupakan penggambaran *database* sistem secara konsep dengan menggunakan *ERD*. *Conceptual Model* dari sistem ini ditunjukkan oleh Gambar 3.12.



Gambar 3.12. *Conceptual Data Model*

3.4.2. *Physical Model* Sistem

Physical Model adalah bentuk fisik dari *Conceptual Model* yang telah di-*mapping*, dimana *Physical Model* ini merupakan penggambaran tabel-tabel yang ada dalam *database* secara fisik. *Physical Model* dari sistem ini ditunjukkan oleh Gambar 3.13.



Gambar 3.13. *Physical Data Model*

3.4.3. *Database* Sistem

Tabel-tabel dalam *database* yang digunakan *software* ini mengambil tabel-tabel penting dari *database* lama. Tabel-tabel ini digunakan untuk menyimpan informasi-informasi yang ada dalam penjadwalan sidang tugas akhir. Informasi-informasi tersebut antara lain informasi mengenai dosen, mahasiswa, proposal mahasiswa, dan sidang tugas akhir mahasiswa. Selain tabel-tabel tersebut, ditambahkan tabel-tabel yang menyimpan informasi mengenai *Fuzzy Relation* dan tabel-tabel yang memiliki peran dalam proses *Genetic Algorithms*. Secara umum, tabel-tabel dalam *database* sistem dapat dikategorikan menjadi empat yaitu tabel sistem informasi, tabel *Fuzzy Relation*, tabel *Genetic Algorithms*, dan tabel lain-lain yang digunakan sebagai tabel bantuan.

Selain tiga kategori di atas, tabel-tabel juga dapat dibagi menjadi tiga berdasarkan sifat data yang disimpan, yaitu tabel *master*, tabel proses, dan tabel *temporary*. Tabel *master* ditandai dengan huruf “M” pada awal nama tabel, dimana data dalam tabel ini merupakan *master* dari data-data yang ada pada tabel lainnya. Tabel proses ditandai dengan huruf “P” pada awal nama tabel, dan

bertujuan untuk menyimpan data-data yang dihasilkan dari sebuah relasi. Tabel *temporary* ditandai dengan huruf "T" pada awal nama tabel, dimana tabel ini hanya digunakan untuk menyimpan data-data yang bersifat sementara yang digunakan dalam proses penyusunan jadwal sidang tugas akhir. Data pada tabel *temporary* ini dapat dihapus ketika sudah tidak dibutuhkan.

3.4.3.1. Tabel Sistem Informasi

Tabel dalam kategori ini bertujuan untuk menyimpan data-data yang digunakan sebagai informasi dalam sistem penjadwalan sidang tugas akhir. Tabel-tabel ini sebagian besar berasal dari tabel-tabel pada *database* lama. Tabel-tabel yang masuk kategori ini ditunjukkan pada Tabel 3.4 sampai Tabel 3.9.

Tabel 3.4. Tabel M_Dosen

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Dosen	Char	5	*	NIP Dosen
Nama_Dosen	VarChar	50		Nama Dosen
Panggilan	VarChar	5		Panggilan untuk Dosen (Bpk/Ibu)
Gelar	Char	2		Gelar Akademik Dosen
Peminatan	VarChar	3		Bidang Peminatan Dosen
Menguji	Boolean			Status Dosen menguji atau tidak
Kuota	Byte			Jumlah Kuota Dosen
DLB	Boolean			Status dosen luar biasa atau tidak
Alamat	VarChar	50		Alamat Dosen
Extension	VarChar	30		Extension Dosen di UK Petra
HP	VarChar	20		Nomor HP Dosen
Telepon_Rumah	VarChar	20		Telepon Rumah Dosen
Telepon_Kantor	VarChar	20		Telepon Kantor Dosen

Tabel 3.5. Tabel M_Mahasiswa

Name	*Type*	*Size*	*Key*	Keterangan
NRP	LongInt		*	NRP Mahasiswa
Nama_Mhs	VarChar	50		Nama_Mhs

Tabel 3.6. Tabel P_Proposal

Name	*Type*	*Size*	*Key*	Keterangan
NRP	LongInt		*	NRP Mahasiswa
Proposal_Ke	Byte		*	Nomor Urut Proposal Mahasiswa
Judul	VarChar	255		Judul Tugas Akhir Mahasiswa
Pembimbing_1	Char	5		NIP Pembimbing 1
Pembimbing_2	Char	5		NIP Pembimbing 2
Peminatan	VarChar	3		Bidang Peminatan Tugas Akhir
Maju_Sidang	Boolean			Status apakah Tugas Akhir disidang atau tidak
Status_Proposal	VarChar	10		Status apakah Tugas Akhir baru diajukan, perpanjang, atau batal
Tahun_Akademik	Char	9		Tahun Akademik Proposal diajukan
Semester	Char	5		Semester Proposal diajukan
Periode	Char	3		Periode Proposal diajukan
Nilai_Maks	VarChar	2		Nilai Maksimum untup Tugas Akhir
Keterangan	VarChar	255		Keterangan lain-lain

Tabel P_Proposal digunakan untuk menyimpan proposal tugas akhir mahasiswa. Dalam *software* ini, proposal tugas akhir seorang mahasiswa ditentukan oleh NRP mahasiswa tersebut dan nomor urut proposal dari mahasiswa tersebut. Untuk lebih jelas, perhatikan ilustrasi berikut.

Misalkan seorang mahasiswa dengan NRP 26401181 mengajukan proposal tugas akhir untuk pertama kali. Dalam *database*, proposal tugas akhir mahasiswa ini diidentifikasikan dengan nilai NRP = 26401181 dan Proposal_Ke = 1. Bila kemudian proposal ini dibatalkan dan mahasiswa tersebut mengajukan proposal baru, maka proposal baru ini diidentifikasikan dengan nilai NRP = 26401181 dan Proposal_Ke = 2. Nomor urut untuk setiap mahasiswa dimulai dari 1 sehingga bila ada mahasiswa lain yang mengajukan proposal, maka Proposal_Ke dari

mahasiswa tersebut dimulai dari 1, bukan melanjutkan nomor dari mahasiswa 26401181.

Tabel 3.7. Tabel P_SidangTA

Name	*Type*	*Size*	*Key*	Keterangan
NRP	LongInt		*	NRP Mahasiswa
Proposal_Ke	Byte		*	Nomor Urut Proposal Mahasiswa
No_Buku	Char	30		Nomor Buku Tugas Akhir
Tanggal	Date		*	Tanggal Sidang
Hari	VarChar	6		Hari Sidang
Sesi	Char	11		Sesi Sidang
Tempat	VarChar	8		Ruang Sidang
Ketua_Penguji	Char	5		NIP Ketua Penguji
Penguji_1	Char	5		NIP Penguji 1
Penguji_2	Char	5		NIP Penguji 2
Tahun_Akademik	Char	9		Tahun Akademik Sidang
Semester	Char	5		Semester Sidang
Periode	Char	3		Periode Sidang
Status_Sidang	VarChar	5		Status sidang lulus atau gagal
Nilai_Angka	Byte			Nilai Tugas Akhir dengan angka
Nilai_Huruf	VarChar	2		Nilai Tugas Akhir dengan huruf
Nilai_B1	Byte			Nilai Pembimbing 1
Nilai_B2	Byte			Nilai_Pembimbing 2
Nilai_KU	Byte			Nilai Ketua Penguji
Nilai_U1	Byte			Nilai Penguji 1
Nilai_U2	Byte			Nilai Penguji 2
Lab	Char	2		Lab yang digunakan

Tabel P_SidangTA digunakan untuk menyimpan informasi mengenai sidang tugas akhir mahasiswa.

Tabel 3.8. Tabel M_Topik

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Topik	Byte		*	Kode Topik
Nama_Topik	VarChar	30		Nama Topik
Keterangan	VarChar	255		Keterangan Topik

Tabel M_Topik digunakan untuk menyimpan topik-topik tugas akhir.

Tabel 3.9. Tabel M_Sesi

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Sesi	Byte		*	Kode Sesi
Waktu_Mulai	Time			Waktu sebuah sesi dimulai
Waktu_Selesai	Time			Waktu sebuah sesi selesai

Tabel M_Sesi digunakan untuk menyimpan informasi mengenai sesi sidang.

3.4.3.2. Tabel *Fuzzy Relation*

Tabel *Fuzzy Relation* digunakan untuk menyimpan nilai-nilai yang digunakan dalam perhitungan dengan menggunakan teori *Fuzzy Relation*. Hasil perhitungan juga disimpan dalam tabel pada kategori ini. Tabel 3.10 sampai Tabel 3.12 menunjukkan tabel-tabel dalam kategori ini.

Tabel 3.10. Tabel P_Expertise

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Dosen	Char	5	*	NIP Dosen
Kode_Topik	Byte		*	Kode Topik
Nilai_Expertise	Float			Nilai keahlian dosen terhadap topik

Tabel P_Expertise digunakan untuk menyimpan nilai *expertise* dosen yang diperoleh dari *input*-an user.

Tabel 3.11. Tabel T_Content

Name	*Type*	*Size*	*Key*	Keterangan
NRP	LongInt		*	NRP Mahasiswa
Proposal_Ke	Byte		*	Nomor Urut Proposal Mahasiswa
Kode_Topik	Byte		*	Kode Topik
Nilai_Content	Float			Nilai content tugas akhir

Tabel T_Content digunakan untuk menyimpan nilai *content* tugas akhir yang diperoleh dari *input*-an user.

Tabel 3.12. Tabel T_Rel_Dosen_Proposal

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Dosen	Char	5	*	NIP Dosen
NRP	LongInt		*	NRP Mahasiswa
Proposal_Ke	Byte		*	Nomor Urut Proposal Mahasiswa
Nilai_Relasi	Float			Tingkat kompetensi dosen
Ketua	Boolean			Status Ketua Penguji
Anggota	Boolean			Status Anggota Penguji

Tabel T_Rel_Dosen_Proposal digunakan untuk menyimpan hasil perhitungan tingkat kompetensi dosen dengan menggunakan *Fuzzy Relation* (Subbab 3.1). Dengan tingkat kompetensi ini, dapat disusun sebuah prioritas mahasiswa dari seorang dosen. *Field* Ketua dan Anggota digunakan untuk menyatakan status dosen dalam sidang berdasarkan gelar dosen. Dosen bergelar S2 atau S3 boleh menjadi ketua penguji, sedangkan dosen dengan gelar S1 hanya boleh menjadi anggota penguji. Jadi, dosen S2 atau S3 memiliki nilai Ketua dan Anggota *true*, sedangkan dosen S1 memiliki nilai Ketua *false* dan Anggota *true*.

3.4.3.3. Tabel *Genetic Algorithms*

Tabel-tabel yang masuk dalam kategori ini adalah tabel-tabel yang secara langsung dipakai dalam proses *Genetic Algorithms*. Tabel-tabel ini menyimpan berbagai keterangan yang dibutuhkan dalam men-*generate* sebuah jadwal sidang

tugas akhir. Tabel 3.13 sampai Tabel 3.17 menjelaskan mengenai tabel-tabel dalam kategori ini.

Tabel 3.13. Tabel T_Jadwal _Sidang

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Tanggal	Byte		*	Kode Tanggal
Tanggal_Sidang	Date			Tanggal Sidang

Tabel T_Jadwal_Sidang digunakan untuk menyimpan tanggal sidang dalam satu periode sidang tugas akhir.

Tabel 3.14. Tabel T_Dosen_Berhalangan

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Dosen	Char	5	*	NIP Dosen
Kode_Tanggal	Byte		*	Kode Tanggal
Kode_Sesi	Byte		*	Kode Sesi

Tabel T_Dosen_Berhalangan digunakan untuk menyimpan jadwal berhalangan dosen. Dari tabel ini, diketahui jadwal bisa dosen.

Tabel 3.15. Tabel T_Maju_Sidang

Name	*Type*	*Size*	*Key*	Keterangan
NRP	LongInt		*	NRP Mahasiswa
Proposal_Ke	Byte		*	Nomor Urut Proposal Mahasiswa
Pembimbing_1	Char	5		NIP Pembimbing 1
Pembimbing_2	Char	5		NIP Pembimbing 2
Ketua_Penguji	Char	5		NIP Ketua Penguji
Penguji_1	Char	5		NIP Penguji 1
Penguji_2	Char	5		NIP Penguji 2
Fired	Boolean			Field bantuan

Tabel T_Maju_Sidang digunakan untuk menyimpan susunan awal mahasiswa dan dosen penguji yang selanjutnya dimasukkan ke dalam gen-gen pada *Genetic Algorithms*. *Field Fired* pada tabel tersebut digunakan untuk menandai apakah *record* tersebut sudah memiliki tempat dalam gen atau belum. Nilai *true* menandakan bahwa data pada *record* tersebut telah dimasukkan ke dalam gen, dan nilai *false* berarti belum dimasukkan.

3.4.3.4. Tabel lain-lain

Tabel-tabel ini merupakan tabel bantuan yang mendukung proses penjadwalan sidang tugas akhir ini. Penjelasan mengenai tabel-tabel ini dapat dilihat pada Tabel 3.16 dan Tabel 3.17.

Tabel 3.16. Tabel T_Dosen_Menguji_Mhs

Name	*Type*	*Size*	*Key*	Keterangan
NRP	LongInt		*	NRP Mahasiswa
Proposal_Ke	Byte		*	Nomor Urut Proposal Mahasiswa
Penguji	Char	5	*	NIP Dosen

Tabel T_Dosen_Menguji_Mhs digunakan untuk menyimpan informasi mengenai mahasiswa yang ingin diuji oleh seorang dosen. Informasi dalam tabel ini digunakan dalam penyusunan dosen penguji untuk seorang mahasiswa. Informasi ini berupa *request* dari dosen untuk memilih mahasiswa yang ingin diujinya.

Tabel 3.17. Tabel T_Dosen_Penguji

Name	*Type*	*Size*	*Key*	Keterangan
Kode_Dosen	Char	5	*	NIP Dosen
Jumlah	Byte			Jumlah berpartisipasi

Tabel T_Dosen_Penguji digunakan untuk menghitung jumlah seorang dosen berpartisipasi dalam sidang tugas akhir baik sebagai pembimbing maupun penguji. Perhitungan ini digunakan untuk pemerataan jadwal berpartisipasi yang menjadi salah satu permasalahan tugas akhir ini.