BAB 6 PENGUJIAN DAN ANALISIS

Bab ini merupakan uraian tentang hasil pengujian yang telah dirancang pada bab perancangan sistem. Lalu membahas hasil dari pengujian yang telah dilakukan. Pada bab ini dilakukan lima kali pengujian, yaitu pengujian berdasarkan ukuran populasi, banyaknya generasi, pengujian nilai *cr* dan *mr*, pengujian nilai bobot dan pengujian konvergensi.

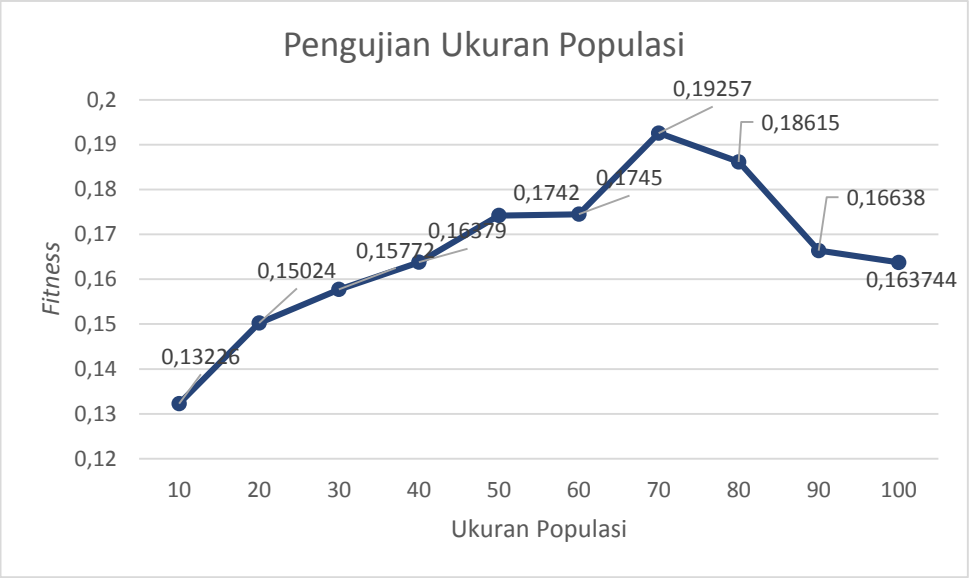
6.1 Pengujian Ukuran Populasi

Proses ini bertujuan untuk mengetahui seberapa bersar ukuran populasi yang bisa didapatan berdasarkan solusi yang paling optimal dengan rata-rata nilai *fitness* terbaik. Pengujian ini dilakukan percobaan sebanyak 10 kali, untuk setiap ukuran populasi dari ukuran populasi 10 sampai dengan 100. Hasil pengujian ukuran populasi ini telah di jelaskan pada Tabel 6.1

Tabel 6.1 Pengujian Ukuran Populasi

Uji Coba Ke-i	Nilai *Fitness* Terbaik Pada Ukuran Populasi									
	10	20	30	40	50	60	70	80	90	100
1	0,139	0,151	0,14	0,155	0,167	0,14	0,179	0,19	0,188	0,182
2	0,108	0,122	0,24	0,167	0,143	0,208	0,179	0,2	0,1633	0,1517
3	0,163	0,114	0,131	0,157	0,166	0,155	0,179	0,152	0,12	0,1658
4	0,113	0,1475	0,163	0,126	0,169	0,152	0,208	0,21	0,147	0,2379
5	0,116	0,143	0,168	0,162	0,133	0,14	0,156	0,1537	0,155	0,10544
6	0,120	0,287	0,128	0,151	0,179	0,176	0,209	0,182	0,167	0,1491
7	0,118	0,137	0,141	0,14	0,15	0,168	0,239	0,15	0,18	0,293
8	0,148	0,15	0,15	0,29	0,206	0,289	0,165	0,208	0,19	0,11
9	0,126	0,135	0,136	0,14	0,186	0,165	0,165	0,23	0,1865	0,136
10	0,17	0,115	0,179	0,149	0,242	0,152	0,244	0,185	0,167	0,1065
Rata-rata	0,132	0,1502	0,157	0,163	0,174	0,174	0,192	0,1861	0,1663	0,1637

Pada Tabel 6.1 nilai rata-rata *fitness* yang didapatkan oleh sistem akan dibuatkan grafik seperti yang ditunjukan pada Gambar 6.1.



Gambar 6.1 Grafik Hasil Pengujian Ukuran Populasi

Berdasarkan table 6.1 dan Gambar 6.1 ukuran populasi mempengaruhi nilai *fitness* dalam setiap percobaan. Hasil percobaan menunjukan semakin kecil ukuran populasi maka semakin kecil juga nilai dari rata-rata *fitness* yang di hasilkan, begitu juga sebaliknya jika nilai populasi tinggi maka nilai dari rata-rata *fitness* tinggi juga.

Pada Gambar 6.1 dapat dilihat bahwa pada ukuran populasi 10 menunjukan nilai dari rata-rata *fitness* yaitu 0,13226. Pada ukuran populasi 10 sampai 60 rata-rata nilai *fitness* terjadi perubahan kenaikan yang signifikan, dan pada ukuran populasi 70 terjadi penaikan puncak nilai *fitness*. Sedangkan pada ukuran populasi 80 sampai 100 terjadi sedikit demi sedikit penurunan. Hal tersebut menunjukan bahwa kemungkinan untuk penambahan ukuran populasi yang lebih besar lagi tidak akan menaikan rata-rata nilai *fitness*. Kesimpulan yang didapat yaitu ukuran populasi terbaik pada pengujian ini adalah 70 populasi.

6.2 Pengujian Jumlah Generasi

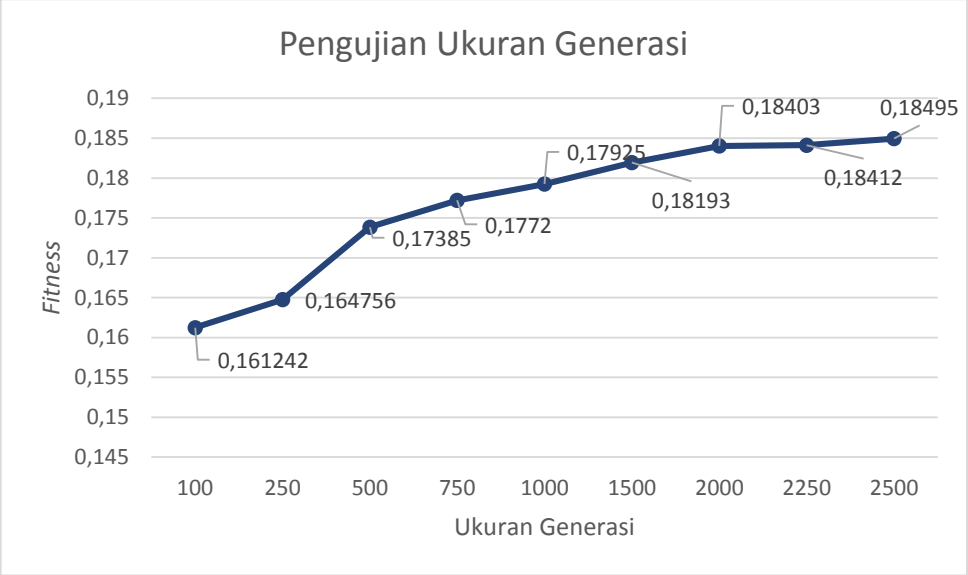
Pengujian jumlah generasi dilakukan untuk mengetahui jumlah generasi terbaik yang paling optimal. Jumlah generasi yang optimal dievaluasi berdasarkan rata-rata nilai *fitness* terbaik. Pengujian ini dilakukan sebanyak 10 kali untuk setiap jumlah generasi dimulai dari generasi ke 100.Hasil pengujian jumlah generasi ditunjukkan pada Tabel 6.2.

Tabel 6.2 Pengujian Jumlah Generasi

Uji Coba Ke-i	Nilai *Fitness* Terbaik pada Generasi									
	100	500	1000	1250	1500	1750	2000	2250	2500	3000
1	0,1621	0,1636	0,18	0,19	0,192	0,193	0,182	0,18	0,1833	0,185
2	0,16285	0,164	0,166	0,18	0,19	0,186	0,1617	0,168	0,188	0,1628
3	0,18757	0,187	0,177	0,164	0,167	0,1676	0,1741	0,18	0,192	0,1875

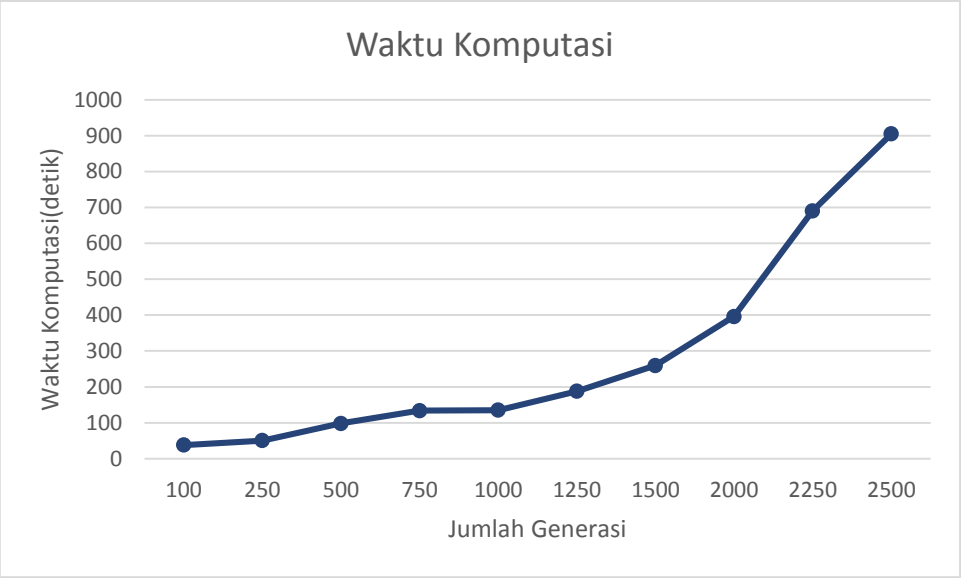
4	0,1408	0,1516	0,1585	0,202	0,22	0,155	0,1605	0,201	0,21	0,1408
5	0,136	0,149	0,171	0,173	0,182	0,208	0,123	0,24	0,201	0,136
6	0,1421	0,15	0,145	0,187	0,1339	0,1514	0,183	0,1622	0,1706	0,1421
7	0,165	0,144	0,188	0,186	0,178	0,1807	0,187	0,181	0,181	0,165
8	0,206	0,207	0,2	0,22	0,19	0,22	0,228	0,141	0,133	0,206
9	0,16	0,17086	0,188	0,104	0,1546	0,189	0,208	0,18	0,1806	0,16
10	0,15	0,1605	0,165	0,166	0,182	0,21	0,233	0,208	0,21	0,15
Rata-rata	0,1612	0,1647	0,1738	0,1772	0,1792	0,1819	0,184	0,1841	0,1849	0,1612

Berdasarkan Tabel 6.2 didapatkan grafik dari rata-rata nilai *fitness.* Grafik hasil pengujian jumlah generasi ditunjukan pada Gambar 6.2.



Gambar 6.2 Grafik Hasil Pengujian Jumlah Generasi

Dapat dilihat bahwa pada ukuran generasi 100 menunjukan nilai dari rata-rata *fitness* yaitu 0,161242 pada Gambar 6.2. Hasil percobaan menunjukan bahwa sedikit jumlah generasi yang digunakan maka semakin nilai rata-rata *fitness* yang didapatkan kecil, dan semakin besar jumlah generasi yang digunakan maka semakin besar pula nilai rata-rata *fitness* yang didapatkan. Pada jumlah generasi mengalami peningkatan rata-rata nilai *fitness* sedikit dan tidak terlalu signifikan. Dalam hal ini memungkinkan terjadinya konvergensi dini, dimana algoritme genetika mengalami nilai *fitness* yang konvergen sebelum menemukan solusi yang paling optimal. Jika hal ini terjadi maka hasil *fitness* selanjutnya tidak akan meningkat secara signifikan. Oleh sebab itu, penulis menggunakan jumlah generasi yaitu sebesar 2.500, karena dengan menambahkan jumlah generasi pada sistem akan memperlama waktu komputasi.



Gambar 6.3 Grafik Waktu Komputasi

6.3 Pengujian Nilai *Cr* dan *Mr*

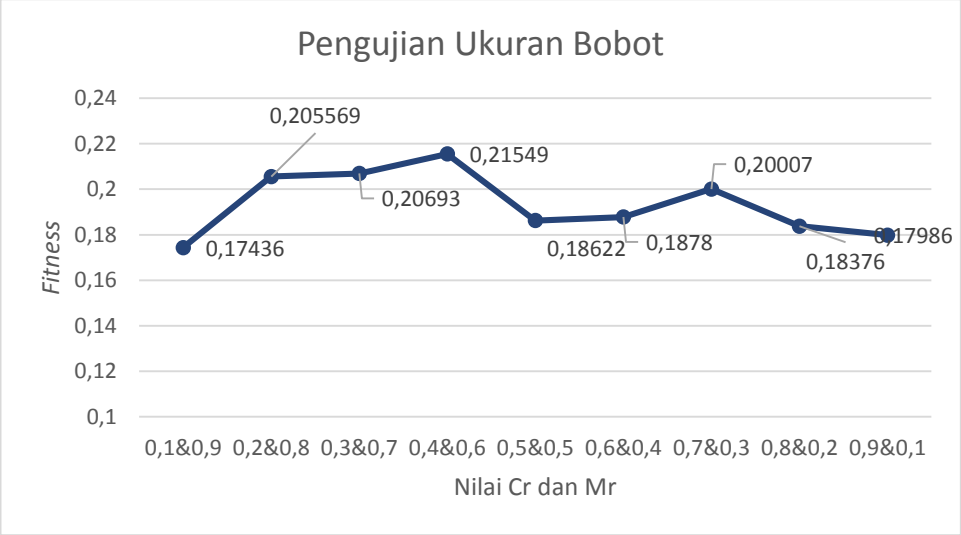
Proses ini menguji nilai *cr* dan *mr* dengan rentang nilai dari 0,1 sampai dengan 0,9 yang mana akan diambil nilai *fitness* terbaik pada saat pengujian berlangsung. Pengujian ini dilakukan sampai 10 kali percobaan pada setiap kombinasi nilai *cr* dan *mr*. Hasil pengujian *cr* dan *mr* ditunjukan pada Tabel 6.3.

Tabel 6.3 Pengujian Nilai *Cr* dan *Mr*

Kombinasi		Nilai *Fitness* Terbaik pada Uji Coba Ke-*i*										Rata-rata
Cr	*Mr*	1	2	3	4	5	6	7	8	9	10	
0,1	0,9	0,163	0,13	0,21	0,15	0,15	0,24	0,14	0,15	0,18	0,207	0,174
0,2	0,8	0,166	0,24	0,23	0,20	0,21	0,18	0,20	0,14	0,20	0,240	0,205
0,3	0,7	0,187	0,17	0,16	0,20	0,14	0,18	0,15	0,16	0,15	0,542	0,206
0,4	0,6	0,204	0,20	0,24	0,23	0,22	0,24	0,21	0,23	0,18	0,183	0,215
0,5	0,5	0,1817	0,17	0,16	0,20	0,151	0,14	0,16	0,14	0,23	0,297	0,186
0,6	0,4	0,243	0,14	0,139	0,169	0,105	0,183	0,165	0,379	0,144	0,211	0,18
0,7	0,3	0,187	0,154	0,173	0,143	0,169	0,155	0,373	0,163	0,24	0,243	0,200
0,8	0,2	0,207	0,176	0,184	0,153	0,157	0,165	0,184	0,19	0,209	0,212	0,183

0,9	0,1	0,207	0,241	0,188	0,152	0,209	0,19	0,19	0,157	0,154	0,11	0,176

Pada Tabel 6.3 dihasilkan nilai dari rata-rata *finess* berdasarkan dari kombinasi nilai *cr* dan *mr* yang memiliki nilai *fitness* terbaik. Gambar 6.3 merupakan hasil proses dari pengujian ini.



Gambar 6.4 Grafik Hasil Pengujian Nilai *Cr* dan *Mr*

Berdasarkan Gambar 6.3 nilai *cr* dan *mr* sangat mempengaruhi rata-rata nilai *fitness* dalam setiap percobaan. Berdasarkan percobaan ini, hasil menunjukan adanya peningkatan pada rata-rata nilai *fitness* pada nilai *cr* 0,1 sampai 0,4, dan pada nilai *mr* 0,9 sampai dengan 0,5. Sedangkan nilai rata-rata *fitness* menurun drastis pada nilai *cr* 0,5 , nilai *mr* 0,5 dan terjadi penaikan sedikit demi sedikit pada *cr* 0,5 sampai 0,7, nilai *mr* 0,5 sampai dengan 0,3. Dari pengujian ini hasil yang didapatkan yaitu nilai *cr* dan *mr* terbaik adalah 0,4 dan 0,6.

6.4 Pengujian Pembobotan Nilai *Constraint*

Pada sub bab pengujian pembobotan ini dilakukan 2 skenario pengujian sesuai dengan rancangan pengujian pada bab 4 yaitu mengubah nilai bobot serentak dan tidak serentak untuk menghasilkan nilai bobot yang optimal.

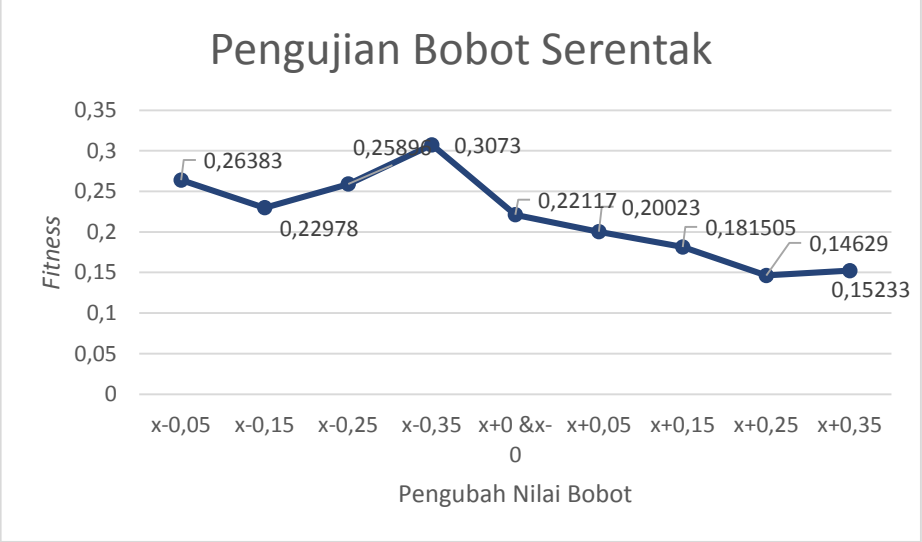
6.4.1 Pengujian Mengubah Nilai Bobot Serentak

Pengujian ini dilakukan dengan mengubah nilai bobot pada *hard constraint* dan *soft constraint* secara bersamaan sesuai dengan variabel pengubahnya. Proses ini dilakukan dengan cara menaikan dan menurunkan secara bertahap sesuai dengan selisih nilai pada setiap perubahannya. Pada pengujian ini dilakukan sebanyak 10 kali percobaan yang diambil berdasarkan nilai rata-rata dari *fitness* yang terbaik. Tabel 6.4 menunjukan hasil pengujian pada proses ini.

Tabel 6.4 Hasil Pengujian Mengubah Nilai Bobot Serentak

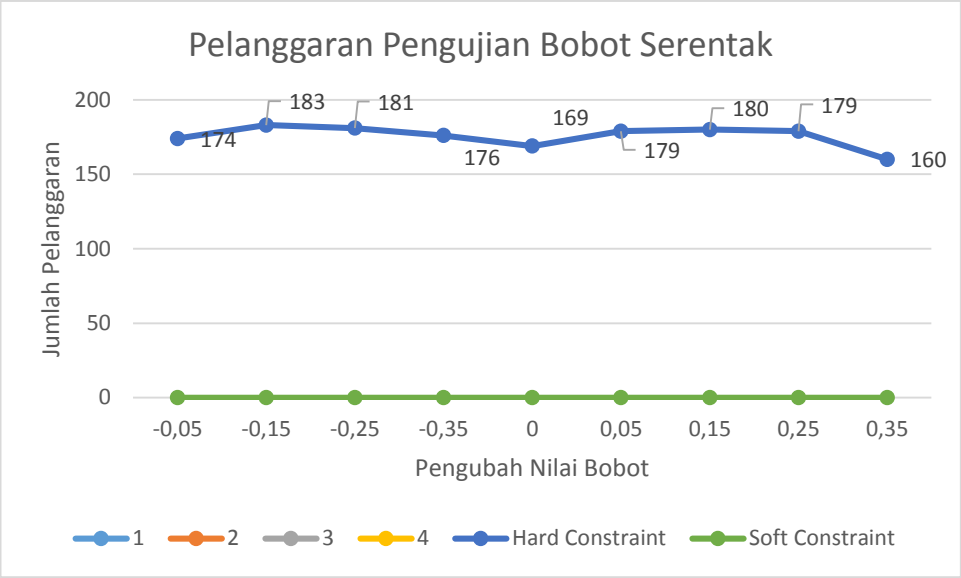
Uji Coba Ke-*i*	Variabel Pengubah Bobot								
	***x*-0,05**	***x*-0,15**	***x*-0,25**	***x*-0,35**	***x*-0,0; *x*+0,0**	***x*+0,05**	***x*+0,15**	***x*+0,25**	***x*+0,35**
1	0,188	0,216	0,273	0,277	0,207	0,1598	0,11	0,124	0,223
2	0,875	0,175	0,3226	0,36	0,183	0,233	0,259	0,113	0,11
3	0,15	0,197	0,246	0,322	0,24	0,138	0,214	0,1133	0,157
4	0,16	0,4205	0,207	0,25	0,168	0,1737	0,186	0,1122	0,1545
5	0,161	0,183	0,27	0,276	0,1577	0,164	0,128	0,1508	0,186
6	0,249	0,24	0,19	0,217	0,185	0,158	0,124	0,15	0,11
7	0,215	0,191	0,378	0,323	0,376	0,165	0,26055	0,1686	0,158
8	0,251	0,202	0,257	0,415	0,171	0,239	0,111	0,195	0,14
9	0,173	0,278	0,213	0,365	0,164	0,3608	0,21	0,148	0,126
10	0,215	0,195	0,233	0,268	0,36	0,211	0,2125	0,188	0,1588
Rata-rata	0,263	0,229	0,2589	0,307	0,2211	0,2002	0,1815	0,1462	0,1523

Berdasarkan pada Tabel 6.4 dihasilkan bentuk grafik untuk nilai rata-rata berdasarkan hasil pengujian ini. Grafik tersebut ditunjukan pada Gambar 6.4.



Gambar 6.5 Hasil Pengujian Pengubahan Nilai Bobot Serentak

Gambar 6.5 yaitu hasil grafik yang ditunjukan di atas berdasarkan pada hasil jumlah pelanggaran yang ditunjukkan menggunakan grafik pada Gambar 6.6.



Gambar 6.6 Jumlah Pelanggaran untuk Pengujian Bobot Serentak

Pada pengujian ini dilakukan dengan cara menaikan dan menurunkan nilai bobot *hard constraint* maupun *soft constraint*. Pada Gambar 6.4 jika dilakukan penurunan nilai bobot maka akan terjai penaikan yang signifikan, pada saat nilai bobot tidak dinaikan dan diturunkan itu berarti nilai dari *fitness* rata-rata akan mencapai puncak yaitu 0,307. Saat penaikan nilai bobot 0,15 terjadi penurunan yang drastis sampai nilai bobot dinaikan 0,35, tetapi pada saat penaikan nilai bobot terjadi penaikan dan penurunan nilai rata-rata *fitness*. Hal ini sama dengan jumlah pelanggaran, yaitu terjadi perubahan naik dan turun untuk jumlah pelanggaran pada *hard constraint*. Sementara untuk *soft constraint* tidak terjadi pelanggaran.

6.4.2 Pengujian Mengubah Nilai Bobot Tidak Serentak

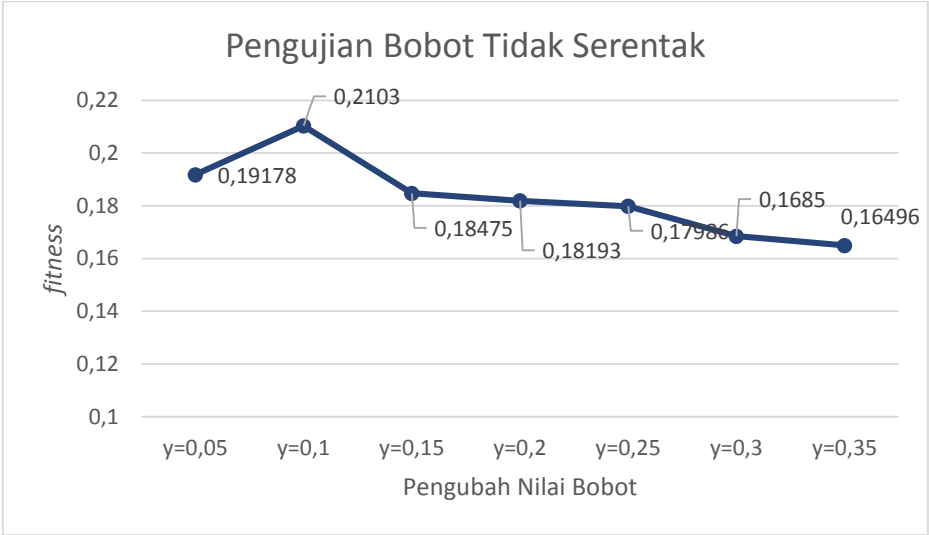
Proses pengujian ini yaitu menaikan nilai bobot pada *hard constraint* dan kemudian menurunkan nilai bobot *soft constraint* sesuai dengan variabel pengubahnya. Proses pengubahan nilai bobot dilakukan dengan cara menaikan dan menurunkannya secara berkala dengan selisih nilai bobot pada setiap nilai pengubahnya. Pada pengujian ini dilakukan sebanyak 10 kali percobaan yang diambil berdasarkan nilai rata-rata dari hasil *fitness* terbaik.Hasil pengujian ini ditunjukan pada Tabel 6.5.

Tabel 6.5 Hasil Pengujian Mengubah Nilai Bobot Tidak Serentak

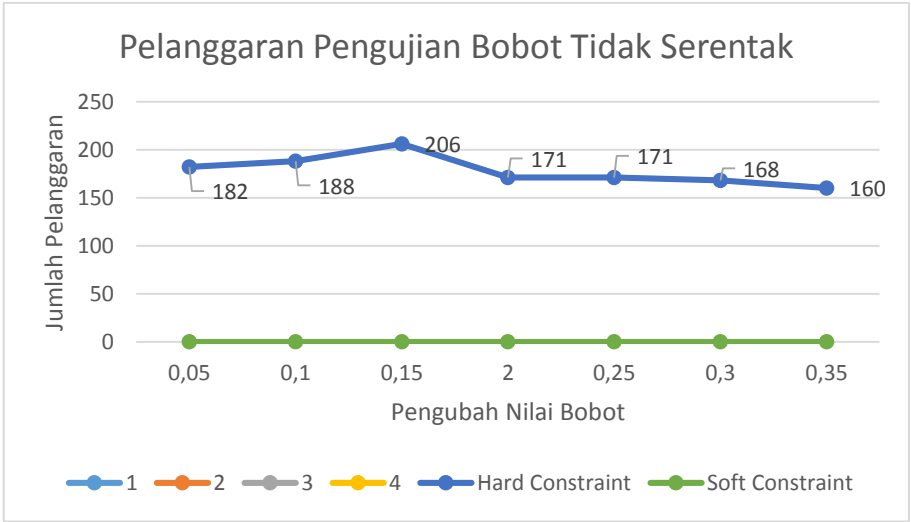
Uji Coba Ke-*i*	Variabel Pengubahan Nilai Bobot						
	***y*=0,05**	***y*=0, 1**	***y*=0, 15**	***y*=0,2**	***y*=0,25**	***y*=0,3**	***y*=0,35**
1	0,161	0,184	0,157	0,194	0,172	0,14	0,1756
2	0,147	0,16	0,153	0,188	0,144	0,17	0,172
3	0,204	0,161	0,148	0,15	0,1512	0,163	0,155
4	0,186	0,275	0,271	0,196	0,135	0,168	0,18

5	0,164	0,198	0,156	0,27	0,21	0,168	0,162
6	0,235	0,18	0,192	0,1523	0,23	0,217	0,224
7	0,282	0,225	0,186	0,16	0,173	0,19	0,156
8	0,14	0,25	0,229	0,14	0,167	0,143	0,165
9	0,234	0,23	0,155	0,16	0,24	0,18	0,16
10	0,164	0,24	0,2	0,209	0,175	0,146	0,1
Rata-rata	0,191	0,210	0,1847	0,1819	0,179	0,168	0,1649

Pada Tabel 6.5 dihasilkan bentuk grafik untuk nilai rata-rata berdasarkan hasil pengujian ini. Grafik tersebut ditunjukan pada Gambar 6.5 dan hasil dari jumlah pelanggaran ditunjjukan pada Gambar 6.8



Gambar 6.7 Hasil Pengujian Mengubah Nilai Bobot Tidak Serentak



Gambar 6.8 Jumlah Pelanggaran untuk Pengujian Bobot Tidak Serentak

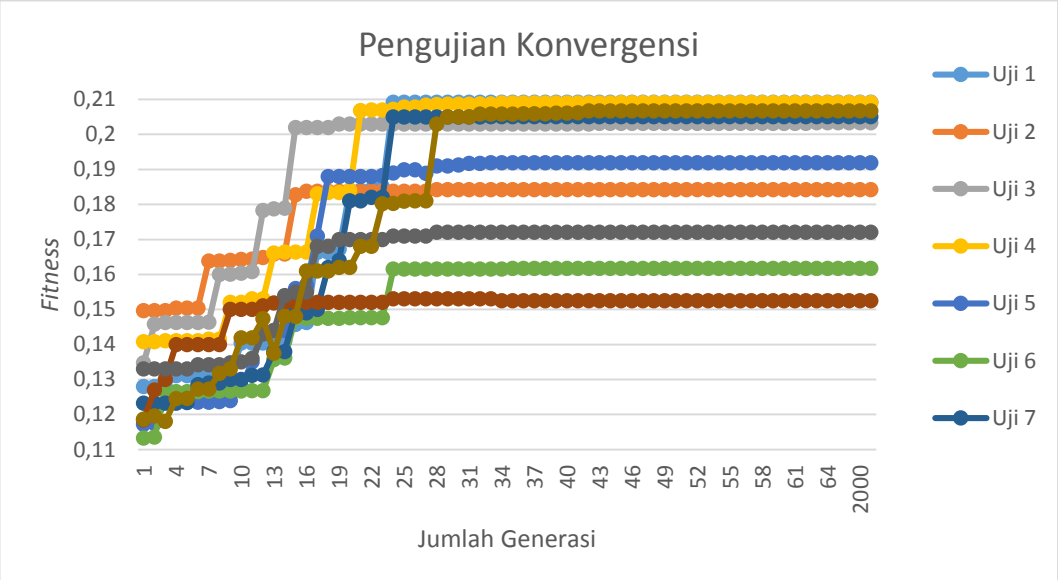
Pengujian ini dilakukan dengan menaikan nilai dari bobot *hard constraint* dan menurunkan nilai dari bobot *soft constraint*. Pada Gambar 6.7 jika semakin besar nilai pengubah variabel *y* itu berarti nilai dari *fitness* akan mengalami penurunan secara perlahan yang mana semakin besar nilai *y* maka nilai dari bobot *hard constraint* akan semakin besar dan nilai dari bobot *soft constraint* akan semakin kecil. Pada *y* = 0,15 sampai dengan *y* = 0,35 nilai *fitness* cenderung menurun seiring dengan bertambahnya nilai dari *hard constraint* dan dengan jumlah pelanggaran *hard constraint* yang cenderung menurun pada y = 0,15 sampai dengan y = 0,35.

Hal ini berarti walaupun terjadi penaikan nilai bobot *hard constraint* dan penurunan nilai *soft constraint* nilai rata-rata *fitness* cenderung mengalami penurunan. Pada saat percobaan sebelumnya didapatkan semakin besar nilai pengubah variabel nilai *fitness* juga akan mengalami penurunan. Maka dapat disimpulkan yaitu pelanggaran pada *hard constraint* memiliki lebih banyak pelanggaran dari pada *soft constraint*.

6.5 Pengujian Konvergensi

Pengujian ini dilakukan untuk mengetahui pada generasi keberapa nilai *fitness* yang dihasilkan sistem mulai menunjukan tidak adanya perubahan. Parameter yang diujikan yaitu mengunakan parameter terbaik dari pengujian yang dilakukan sebelumnya. Berdasarkan hasil yang didapatkan seperti pada Gambar 6.8 generasi tertinggi yaitu 0,2092 pada generasi ke 24 percobaan ke-1.

Pengujian konvergensi ini menunjukan bahwa pencarian solusi terjebak pada optimum lokal karena nilai *fitness* tidak mengalami perubahan pada generasi ke-43 dan seterusnya. Konvergensi dini sering terjadi dikarenakan populasi solusi terjebak pada optimum lokal dan kecepatan pencarian pada titik solusi akan menurun jika mendekati titik optimum.



Gambar 6.9 Pengujian Konvergensi

6.6 Analisis Global dari Keseluruhan Hasil Pengujian

Berdasarkan perancangan, implementasi, pengujian dan analisis diketahui bahwa algoritme genetika dapat digunakan untuk penjadwalan bimbingan skripsi. Beberapa tahap dalam mengimplementasikan sistem ini adalah sebagai berikut:

1. Inisialisasi kromosom, yaitu menentukan individu awal secara *random* sesuai dengan jumlah *popsize* yang di tentukan oleh *user* dengan panjang kromosom yang di tentukan. Dalam penelitian ini panjang kromosom sepanjang 5. Kromosom ini merepresentasikan jadwal bimbingan mahasiswa yang akan diisi nilai dengan kode bimbingan yang nantinya di sesuaikan dengan jadwal kode mengajar dosen dan kode perkuliahan mahasiswa, apakah terdapat bentrok atau tidak.
2. Reproduksi terbagi atas *crossover* dan mutasi. Proses pada reproduksi akan menghasilkan keturunan (*offspring*).
3. Evaluasi *fitness* yaitu menghitung nilai *fitness* dari seluruh populasi gabungan antara induk dan anak dan akan dilakukan perhitungan pelanggaran yang terjadi.
4. Seleksi yaitu tahap akhir dalam proses utama algoritme genetika. Tahap seleksi memilih individu yang memiliki nilai tertinggi pada *fitness* yang kemudian akan dilanjutkan ke generasi selanjutnya sampai dengan generasi yang ditentukan atau sampai didapatkan solusi yang optimal.

Sistem penjadwalan bimbingan skripsi menggunakan algoritme genetika dijalankan dengan beberapa parameter yang optimal yang dihasilkan oleh sistem dari beberapa pengujian sebelumnya, yaitu ukuran populasi yang terbaik dihasilkan adalah 70, jumlah generasi terbaik yang dihasilkan adalah 2.500, nilai *cr* dan *mr* yang terbaik dihasilkan adalah 0,4, dan 0,6. Dengan parameter yang diperoleh didapatkan nilai *fitness* sebesar 1,0305 dengan jumlah sukses sebanyak 133 individu dan jumlah gagal sebanyak 117 individu.

Berdasarkan dari hasil solusi terbaik yang didapatkan dari pengujian yang dilakukan, masih terdapat pelanggaran yang terjadi pada jadwal yang dihasilkan sistem. Detail pelanggaran aturan yang terjadi pada jadwal ditunjukkan pada Tabel 6.6. Jadwal yang dihasilkan oleh sistem ini masih belum optimal namun pada aturan yang ke-4 sudah tidak terjadi pelanggaran, meskipun pada aturan 1, 2, dan 3 masih terdapat pelanggaran pada *hard constraint*.

Tabel 6.6 Detail Pelanggaran Aturan pada Jadwal yang Dihasilkan Sistem

No	*Constraint*	Jumlah Pelanggaran
1	Terdapat bentrok antara jadwal perkuliahan mahasiswa dengan jadwal mengajar dosen 1	78
2	Terdapat bentrok antara jadwal perkuliahan mahasiswa dengan jadwal mengajar dosen 2	55

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
TOTAL PELANGGARAN		134
NILAI *FITNESS*		1,0305

Penulis melakukan perbandingan jadwal bimbingan skripsi yang telah dibuat secara manual. Jadwal yang telah dibuat secara manual tidak terdapat pelanggaran *hard constraint* maupun *soft constraint* sistem manual ini. Detail pelanggaran pada jadwal yang dibuat scara manual ditampilkan pada Tabel 6.7.

Tabel 6.7 Detail Pelanggaran Aturan pada Jadwal Manual

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

Dari hasil yang dihasilkan sistem masih belum dapat mencapai optimal karena beberapa alasan. Pertama menjadwalkan untuk 250 mahasiswa dan 60 dosen. Sedangkan nilai kromosom dibangkitkan secara acak tidak menjamin semua individu mempunyai susunan kromosom yang baik. Kedua pada pengujian konvergensi algoritme genetika tidak dapat mengekplorasi lebih banyak dikarenakan populasi yang ada mempunyai kemiripan yang tinggi sehingga terjadi konvergensi. Jika ditinjau dari total pelanggaran aturan, jadwal manual lebih baik karena tidak ada sama sekali pelanggaran yang terjadi dan memiliki nilai *fitness* maksimal.

Untuk dapat meningkatkan nilai *fitness*, pertama yaitu sistem ini membutuhkan slot sebanyak 250, dimana jika sistem memproses jadwal

bimbingan kurang dari jumlah slot yang ada, maka kemungkinan penyebaran jadwal bimbingan pada sistem akan lebih baik dilakukan, sehingga lebih sedikit pinalti yang dilanggar dan nilai *fitness* akan lebih tinggi.

Kedua pada proses *random mutation* dapat dilakukan perubahan pada sejumlah gen misalkan 50% dari jumlah panjang kromosom, yang nantinya gen tersebut dapat lebih baik dari sebelumnya. Dengan cara tersebut dapat juga meningkatkan nilai *fitness* terhadap individu yang ada.

Dibalik alasan sulitnya menemukan solusi optimal, sistem telah menghasilkan jadwal bimbingan yang mendekati optimal. Keberhasilan sistem ini dapat ditinjau dari waktu komputasi ± 15 menit untuk mendapatkan hasil yang optimal dan mahasiswa yang ingin melakukan bimbingan mendapatkan jadwal yang pasti untuk melakukan jadwal bimbingan.