BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Masalah

Berdasarkan penelitian yang telah dilakukan sebelumnya mengenai penyelesaian permainan *puzzle* kakuro dengan menggunakan algoritma *backtracking*, maka ditemukan masalah yang akan diperbaiki oleh algoritma lain pada penelitian ini yaitu dengan menggunakan algoritma genetika.

Mengacu pada penelitian sebelumnya yang menggunakan algoritma *backtracking* untuk menyelesaikan permainan *puzzle* kakuro terdapat masalah dalam penyelesaian permainan dengan ukuran yang besar yaitu dengan ukuran 14 x 14. Hasil dari penelitian tersebut untuk menyelesaikan permainan *puzzle* kakuro dengan ukuran 14 x 14 memerlukan waktu lebih dari 20 menit. Hal ini disebabkan oleh terjadinya rekursif sebanyak lebih dari 8 juta kali [**1**].

Untuk mengatasi permasalahan tersebut maka diperlukan solusi untuk menyelesaikan permainan *puzzle* kakuro yang berukuran besar dengan waktu yang lebih cepat. Algoritma genetika dapat menjadi pilihan karena memiliki keuntungan dalam menemukan solusi pada ruang masalah yang cukup besar dan solusi yang diharapkan tidak harus paling optimal tetapi cukup bisa diterima.

3.1.1 Analisis Permainan *Puzzle* Kakuro

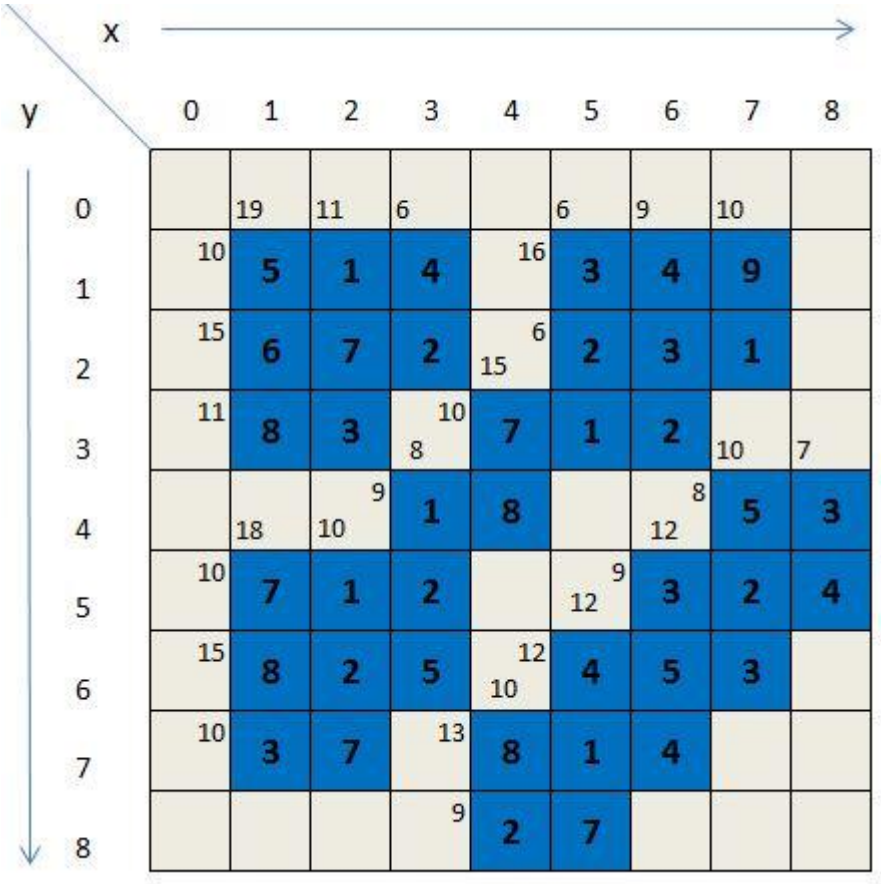
Analisis permainan yang akan diteliti pada permainan *puzzle* kakuro adalah dengan membangkitkan soal permainan (*generator*) serta menyelesaikan jawaban (*solver*). Pada dasarnya permainan *puzzle* kakuro ini disetiap soal maupun jawaban akan saling terkait satu sama lainnya dan mempunyai nilai pasti disetiap jawabannya. Pada penelitian ini algoritma genetika akan diterapkan untuk menyelesaikan jawaban permainan *puzzle* kakuro.

Pada gambar 3.1 merupakan keadaan awal dari permainan *puzzle* kakuro berukuran 9 x 9 sebelum ditentukan nilai soal pada setiap lajurnya. Setiap kotak akan direpresentasikan kedalam koordinat x dan y. Kotak yang berwarna biru merupakan kotak yang berisi nilai jawaban. Sedangkan kotak yang berwarna abu -

abu akan menjadi tempat nilai soal yang akan menjadi acuan dalam mengisi kotak jawaban. Pada penelitian ini kotak permainan akan dibuat terlebih dahulu, kemudian nilai soal akan dibangkitkan secara acak. Kotak permainan dibuat dengan menggunakan fasilitas pembuat tabel dalam visual studio yaitu *DataGridView.*

3.1.1.1 Analisis Kotak Permainan

Kotak permainan yang ada pada gambar 3.1 merupakan contoh kotak permainan berukuran 9 x 9 yang akan digunakan pada proses analisis Setiap kotak akan direpresentasikan kedalam koordinat x dan y. Kotak yang berwarna biru merupakan kotak yang berisi nilai jawaban. Sedangkan kotak yang berwarna abu - abu akan menjadi tempat nilai soal yang akan menjadi acuan dalam mengisi kotak jawaban.



Gambar 3. 1 Contoh Kotak Permainan Yang Akan Digunakan Pada Tahap Analisis

Lajur dalam permainan *puzzle* kakuro dibagi menjadi 2 yaitu lajur horizontal dan lajur vertikal. Berikut adalah lajur – lajur horizontal yang terdapat pada kotak permainan :

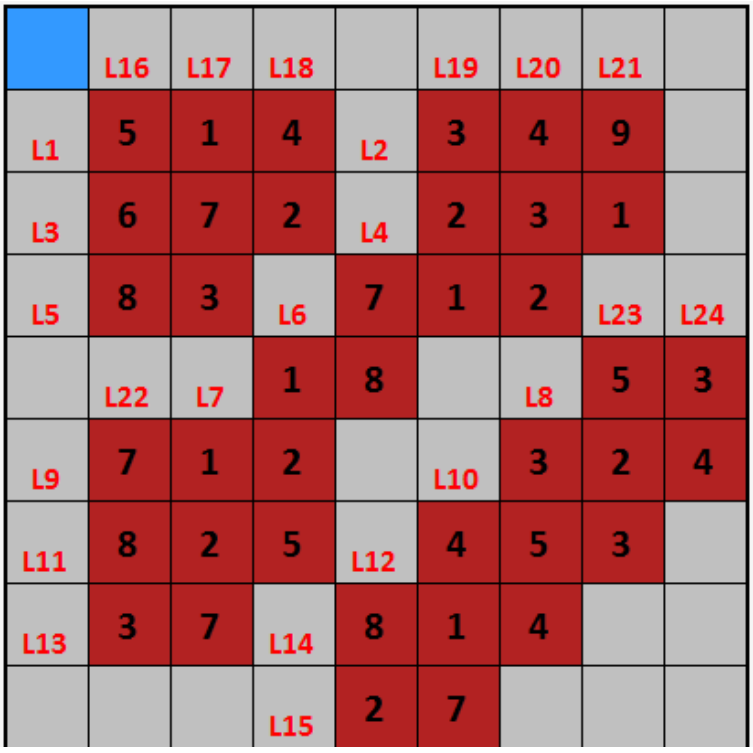
a. Lajur 1 : Koordinat (1,1) sampai (3,1).
b. Lajur 2 : Koordinat (5,1) sampai (7,1).
c. Lajur 3 : Koordinat (1,2) sampai (3,2).
d. Lajur 4 : Koordinat (5,2) sampai (7,2).
e. Lajur 5 : Koordinat (1,3) sampai (2,3).
f. Lajur 6 : Koordinat (4,3) sampai (6,3).
g. Lajur 7 : Koordinat (3,4) sampai (4,4).
h. Lajur 8 : Koordinat (7,4) sampai (8,4).
i. Lajur 9 : Koordinat (1,5) sampai (3,5).
j. Lajur 10 : Koordinat (6,5) sampai (8,5).
k. Lajur 11: Koordinat (1,6) sampai (3,6).
l. Lajur 12 : Koordinat (5,6) sampai (7,6).
m. Lajur 13 : Koordinat (1,7) sampai (2,7).
n. Lajur 14 : Koordinat (4,7) sampai (6,7).
o. Lajur 15 : Koordinat (4,8) sampai (5,8).

Berikut adalah lajur – lajur vertikal yang terdapat pada kotak permainan :

a. Lajur 1 : Koordinat (1,1) sampai (1,3).
b. Lajur 2 : Koordinat (2,1) sampai (2,3).
c. Lajur 3 : Koordinat (3,1) sampai (3,2).
d. Lajur 4 : Koordinat (5,1) sampai (5,3).
e. Lajur 5 : Koordinat (6,1) sampai (6,3).
f. Lajur 6 : Koordinat (7,1) sampai (7,2).
g. Lajur 7 : Koordinat (4,3) sampai (4,4).
h. Lajur 8 : Koordinat (3,4) sampai (3,6).
i. Lajur 9 : Koordinat (7,4) sampai (7,6).
j. Lajur 10 : Koordinat (8,4) sampai (8,5).
k. Lajur 11: Koordinat (1,5) sampai (1,7).

l. Lajur 12 : Koordinat (2,5) sampai (2,7).
m. Lajur 13 : Koordinat (6,5) sampai (6,7).
n. Lajur 14 : Koordinat (5,6) sampai (5,8).
o. Lajur 15 : Koordinat (4,7) sampai (4,8).

Pada gambar 3.2 merupakan contoh kotak permainan yang akan digunakan di dalam sistem. Kotak yang berwarna merah merupakan kotak yang berisi nilai jawaban. Sedangkan kotak yang berwarna abu - abu akan menjadi tempat nilai soal yang akan menjadi acuan dalam mengisi kotak jawaban. Kotak permainan dibuat dengan menggunakan fasilitas pembuat tabel dalam visual studio yaitu *DataGridView*. Dikarenakan setiap *cell* dari *DataGridView* tidak dapat menampikan nilai soal seperti pada contoh di gambar 3.1 maka nilai soal akan direpresentasikan oleh simbol L, contoh nya untuk merepresentasikan nilai lajur 1 maka pada kotak permainan akan ditampilkan L1. Sedangkan nilai soal dari lajur tersebut akan ditampilkan di sebelah kotak permainan.

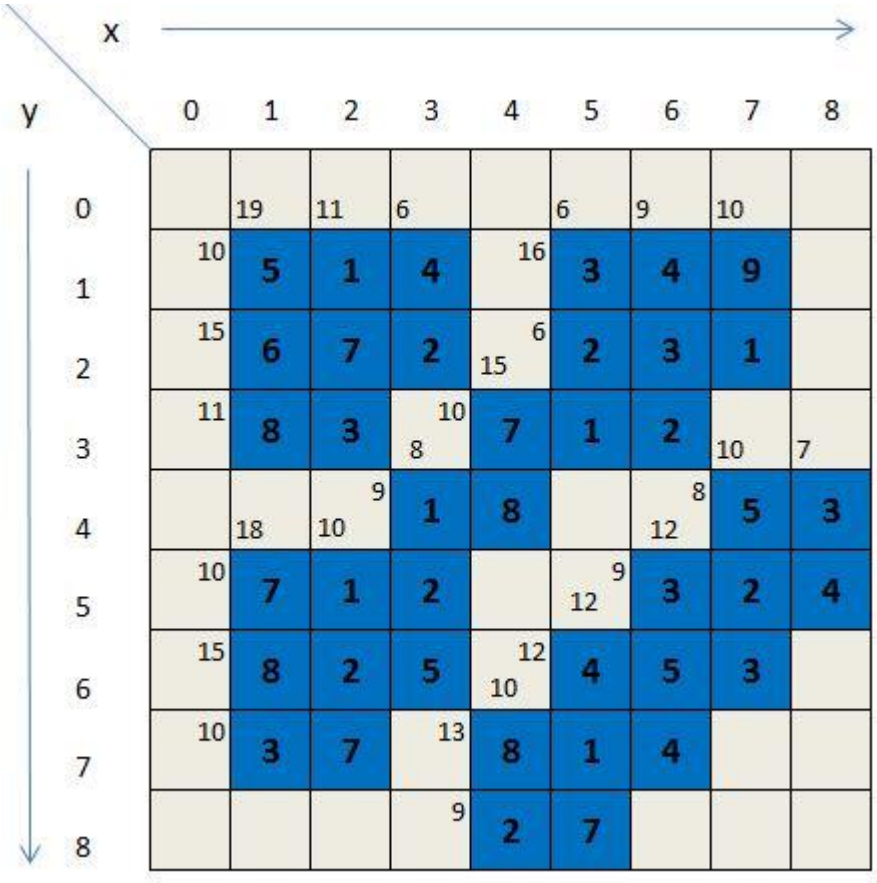


Gambar 3. 2 Contoh Kotak Permainan Yang Akan Digunakan Pada Sistem

3.1.1.2 Analisis Proses Pembangkitan Nilai Soal

Pada proses pembangkitan nilai soal, akan dilakukan pembangkitan bilangan secara acak yang akan menghasilkan nilai jawaban pada setiap lajurnya. Hasil dari pembangkitan nilai jawaban tersebut harus memenuhi aturan dimana tidak ada nilai yang sama pada satu lajur, kemudian nilai – nilai tersebut akan diakumulasikan setiap lajurnya dan menjadi nilai soal. Daftar dari nilai soal dapat dilihat pada Tabel 3.1.

Untuk contoh dari *puzzle* kakuro yang telah dibangkitkan nilai soal dan nilai jawaban dapat dilihat pada gambar 3.3.



Gambar 3.3 Hasil Proses Pembangkitan Nilai Soal dan Nilai Jawaban

Nilai soal ini akan digunakan acuan pada saat algoritma genetika diimplementasikan untuk menyelesaikan permainan *puzzle* kakuro. Sedangkan

nilai jawaban yang sudah didapatkan akan menjadi bukti bahwa *puzzle* dengan nilai soal tersebut memiliki solusi atau dengan kata lain dapat diselesaikan.

3.1.1.3 Proses Penyelesaian Permainan

Pada proses ini, kotak permainan yang telah dibangkitkan nilai soalnya akan diselesaikan dengan menerapkan algoritma genetika. Proses yang akan dilakukan oleh algoritma genetika adalah mengisi semua kotak kosong dengan bilangan acak dan kemudian melakukan pengecekan apakah angka – angka yang terisi sudah sesuai dengan aturan dan sesuai dengan nilai soal. Jika semua aturan sudah terpenuhi maka proses selesai dan *puzzle* berhasil diselesaikan.

3.1.2 Analisis Masukan

Analisis masukan yang dilakukan dalam penelitian ini adalah menganalisis nilai masukan yang akan digunakan pada permainan *puzzle* kakuro. Masukan tersebut berupa nilai (angka), yaitu dari angka 1 hingga 9.

Berdasarkan aturan permainan *puzzle* kakuro yang tidak memperbolehkan adanya angka yang sama pada satu lajur, maka dapat dihasilkan kombinasi nilai masukan dari lajur dengan 2 kotak permainan yang memenuhi aturan permainan. Kombinasi nilai soal dapat dilihat pada tabel 3.1.

Tabel 3.1 Nilai Masukan 2 Kotak

NILAI SOAL	NILAI KOMBINASI
3	1-2
4	1-3
5	1-4; 2-3
6	1-5; 2-4
7	1-6; 2-5; 3-4
8	1-7; 2-6; 3-5
9	1-8; 2-7; 3-6; 4-5
10	1-9; 2-8; 3-7; 4-6
11	2-9; 3-8; 4-7; 5-6
12	3-9; 4-8; 5-7

13	4-9; 5-8; 6-7
14	5-9; 6-8
15	6-9; 7-8
16	7-9
17	8-9

Berdasarkan aturan permainan *puzzle* kakuro yang tidak memperbolehkan adanya angka yang sama pada satu lajur, maka dapat dihasilkan kombinasi nilai masukan dari lajur dengan 3 kotak permainan yang memenuhi aturan permainan. Kombinasi nilai soal dapat dilihat pada tabel 3.2.

Tabel 3.2 Nilai Masukan 3 Kotak

NILAI SOAL	NILAI KOMBINASI
6	1-2-3
7	1-2-4
8	1-2-5; 1-3-4
9	1-2-6; 1-3-5; 2-3-4
10	1-2-7; 1-3-6; 1-4-5; 2-3-5
11	1-2-8; 1-3-7; 1-4-6; 2-3-6; 2-4-5
12	1-2-9; 1-3-8; 1-4-7; 1-5-6; 2-3-7; 2-4-6; 3-4-5
13	1-3-9; 1-4-8; 1-5-7; 2-3-8; 2-4-7; 2-5-6; 3-4-6
14	1-4-9; 1-5-8; 1-6-7; 2-3-9; 2-4-8; 2-5-7; 3-4-7; 3-5-6
15	1-5-9; 1-6-8; 2-4-9; 2-5-8; 2-6-7; 3-4-8; 3-5-7; 4-5-6
16	1-6-9; 1-7-8; 2-5-9; 2-6-8; 3-4-9; 3-5-8; 3-6-7; 4-5-7
17	1-7-9; 2-6-9; 2-7-8; 3-5-9; 3-6-8; 4-5-8; 4-6-7

18	1-8-9; 2-7-9; 3-6-9; 3-7-8; 4-5-9; 4-6-8; 5-6-7
19	2-8-9; 3-7-9; 4-6-9; 4-7-8; 5-6-8
20	3-8-9; 4-7-9; 5-6-9; 5-7-8
21	4-8-9; 5-7-9; 6-7-8
22	5-8-9; 6-7-9
23	6-8-9
24	7-8-9

3.1.3 Analisis Metode

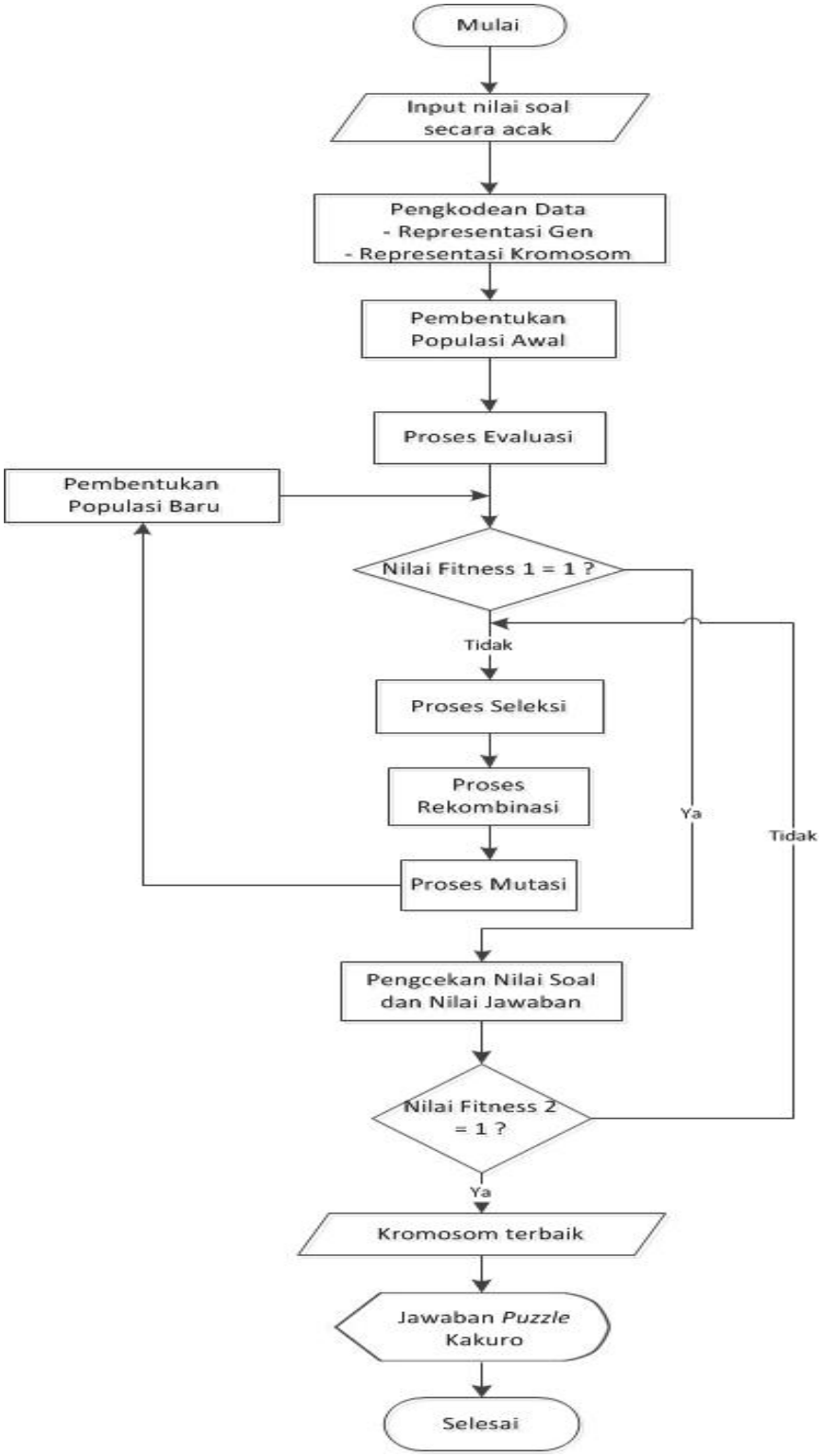
Analisis metode merupakan analisis penerapan metode atau suatu algoritma dalam menyelesaikan suatu permasalahan yang ada. Metode dalam solusi pemecahan masalah dalam penelitian ini adalah Algoritma Genetika.

3.1.3.1 Tahapan Algoritma Genetika

Terdapat rangkaian proses pada algoritma genetika yang dilakukan untuk menyelesaikan permainan *puzzle* kakuro. Setelah nilai soal berhasil dibangkitkan, maka algoritma genetika akan melakukan proses – proses untuk mencari nilai jawaban yang memenuhi aturan pada permainan *puzzle* kakuro. Berikut adalah tahap – tahap implementasi algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro :

1. Pengkodean data (Representasi data).
2. Pembentukan populasi awal.
3. Proses evaluasi (Menghitung nilai *fitness*).
4. Proses Seleksi.
5. Proses Pengecekan Nilai Jawaban dan Nilai Soal.
6. Proses Rekombinasi (*Crossover*).
7. Proses Mutasi.
8. Pembentukan Populasi Baru
9. Hasil Proses Algoritma Genetika.

Tahapan algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro dapat dilihat pada gambar 3.4 :



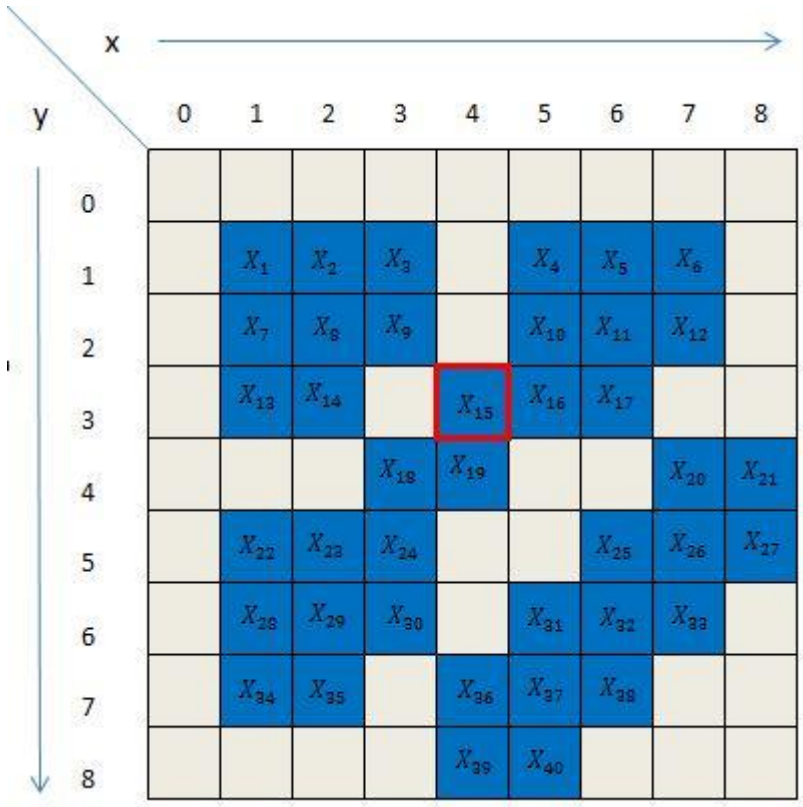
Gambar 3.4 Flowchart Implementasi Algoritma Genetika Untuk Menyelesaikan Permainan *Puzzle* Kakuro

3.1.3.1.1 Pengkodean Data

Pengkodean data merupakan penggunaan data-data yang akan digunakan sebagai nilai variabel yang akan digunakan dalam proses algoritma genetika. Pengkodean tersebut yaitu representasi gen dan representasi kromosom. Berikut adalah penjelasan representasi gen dan representasi kromosom :

1. Representasi Gen

   Yang akan menjadi gen pada penelitian ini adalah setiap kotak yang digunakan untuk mengisi jawaban yang terdapat pada permainan *puzzle* kakuro. Bentuk dari nilai gen tersebut adalah nilai integer. Untuk *allele* yang digunakan adalah bilangan bulat dari yang memiliki nilai 1 sampai 9. Gambaran dari representasi gen dan *allele* dapat dilihat pada gambar 3.5 :



Gambar 3.5 Representasi Gen, Allele dan Kromosom

Berdasarkan pada gambar 3.5, terdapat satu kotak bergaris merah pada koordinat (4,3) yang menggambarkan bahwa satu kotak tersebut

merepresentasikan satu gen, sedangkan variabel X_{15} merupakan representasi dari *allele* yang akan berisikan nilai 1 sampai 9.

2. Representasi Kromosom

   Yang akan menjadi kromosom pada penelitian ini adalah gabungan dari seluruh kotak masukan yang digunakan untuk mengisi jawaban yang terdapat pada permainan *puzzle* kakuro. Panjang dari kromosom sama dengan jumlah kotak masukan tersebut. Pada contoh permainan *puzzle* kakuro yang terdapat pada gambar 3.5 yang berukuran 9 x 9, memiliki panjang kromosom 40. Setiap kromosom akan direpresentasikan oleh sebuah array.

3.1.3.1.2 Pembentukan Populasi Awal

Proses pembentukan populasi awal yang digunakan pada penelitian ini menggunakan metode *random search*. Proses tersebut dilakukan dengan cara memberikan nilai gen – gen awal dengan nilai dari angka 1 sampai 9. Populasi yang digunakan pada penelitian ini berjumlah 10 populasi. Untuk panjang kromosom yang akan digunakan tergantung dari ukuran permainan *puzzle*, akan tetapi konsepnya tetap sama. Yang akan digunakan sebagai contoh adalah permainan *puzzle* kakuro yang dapat dilihat pada gambar 3.2 dengan panjang kromosom 40. Sedangkan pada ukuran 14 x 14 yang akan diimplementasikan pada sistem panjang kromosomnya sebesar 104.

Berikut adalah hasil dari pembentukan populasi awal yang dapat dilihat pada tabel 3.3 :

Tabel 3.3 Pembentukan Populasi Awal

KROMOSOM	NILAI GEN
1	6, 4, 4, 5, 5, 2, 3, 3, 8, 8, 3, 4, 5, 1, 5, 6, 1, 8, 6, 5, 1, 8, 2, 8, 8, 7, 9, 8, 3, 7, 8, 4, 2, 6, 3, 2, 9, 3, 3, 8,
2	3, 6, 4, 9, 8, 4, 7, 1, 6, 1, 7, 8, 1, 2, 1, 6, 4, 4, 5, 2, 2, 3, 5, 6, 3, 8, 5, 1, 8, 5, 1, 2, 4, 9, 8, 7, 2, 8, 9, 1,
3	4, 5, 3, 5, 6, 9, 5, 7, 5, 2, 8, 4, 1, 9, 5, 1, 4, 1, 1, 1, 8, 6, 6, 8, 3, 6, 8, 1, 3, 3, 9, 5, 7, 1, 1, 5, 9, 4, 2, 5,
4	9, 1, 5, 5, 2, 4, 9, 1, 7, 5, 7, 7, 1, 9, 7, 4, 6, 3, 7, 9, 8, 6, 7, 3, 2, 8, 1, 3, 2, 5, 9, 5, 1, 8, 1, 4, 4, 1, 4, 5,

5	7, 7, 9, 6, 9, 7, 9, 7, 4, 5, 9, 4, 1, 1, 1, 1, 2, 3, 3, 3, 9, 2, 8, 1, 1, 6, 8, 9, 4, 1, 9, 9, 1, 6, 1, 7, 1, 6, 4, 8,
6	2, 3, 5, 8, 5, 2, 4, 5, 2, 8, 1, 7, 2, 2, 9, 3, 3, 4, 4, 1, 3, 5, 7, 4, 3, 5, 2, 9, 9, 4, 7, 2, 3, 2, 5, 5, 4, 4, 8, 1,
7	8, 3, 9, 9, 6, 9, 2, 8, 9, 4, 6, 2, 9, 8, 6, 6, 9, 5, 1, 6, 6, 4, 3, 6, 5, 6, 4, 2, 1, 4, 7, 2, 8, 8, 1, 4, 2, 4, 4, 8,
8	4, 5, 1, 2, 3, 9, 9, 2, 2, 7, 9, 7, 2, 8, 5, 7, 5, 4, 5, 9, 8, 7, 1, 3, 5, 2, 4, 4, 9, 2, 3, 6, 6, 4, 8, 8, 6, 6, 7, 6,
9	4, 8, 9, 8, 6, 1, 5, 5, 8, 2, 6, 6, 5, 7, 1, 7, 6, 1, 8, 6, 2, 4, 9, 7, 8, 1, 6, 7, 7, 3, 9, 7, 3, 2, 5, 8, 8, 9, 2, 1,
10	5, 3, 5, 8, 3, 3, 8, 4, 4, 5, 2, 7, 9, 3, 8, 7, 3, 1, 8, 7, 7, 7, 2, 1, 8, 3, 8, 5, 5, 8, 6, 8, 6, 7, 9, 6, 7, 6, 2, 7,

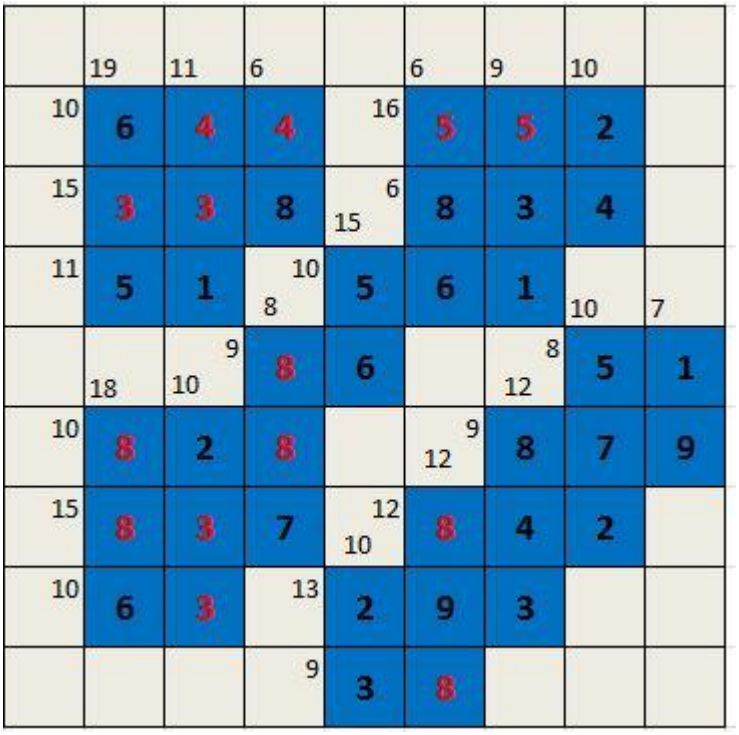
3.1.3.1.3 Proses Evaluasi (Menghitung Nilai *Fitness*)

Setelah populasi awal dibentuk maka proses selanjutnya adalah menghitung nilai *fitness* dari masing – masing kromosom. Nilai fitness yang dinyatakan sebagai solusi dalam penelitian ini bernilai 2, kromosom dengan nilai *fitness* 2 adalah kromosom yang sudah memenuhi aturan tidak boleh ada nilai yang sama pada satu lajur, dan dapat dikatakan menyelesaikan permainan. Yang pertama dilakukan adalah memasukkan setiap kromosom pada *puzzle* permainan. Setelah itu akan dilakukan proses penghitungan nilai fitness yang akan menggunakan persamaan 2.1, 2.2 dan 2.3 yang telah dijelaskan pada bab 2.

Pada gambar 3.6 kromosom 1 yang telah didefinisikan sebelumnya dimasukkan kedalam *puzzle*. Di dalam *puzzle* terdapat angka yang berwarna merah yang menandakan bahwa nilai dari gen tersebut tidak memenuhi aturan permainan *puzzle* kakuro karena terdapat angka yang sama pada satu lajur baik horizontal ataupun vertikal. Dari hasil penempatan kromosom 1 terdapat 20 gen yang tidak memenuhi aturan permainan.

Selanjutnya setiap kromosom akan dihitung nilai *fitness* nya. Jika pada perhitungan dengan menggunakan persamaan 2.1 menghasilkan nilai tidak sama dengan 1, maka nilai *fitness* yang dihasilkan oleh perhitungan yang ada pada persamaan 2.2 akan diberi nilai 0. Hal ini dikarenakan proses perhitungan lajur yang benar hanya akan dilakukan jika semua nilai jawaban pada kromosom sudah memenuhi aturan tidak ada nilai yang sama pada satu lajur yang ditunjukkan

dengan nilai *fitness* 1 dari persamaan 2.1. Berikut adalah perhitungan dari nilai *fitness* kromosom 1 hingga kromosom 10 :



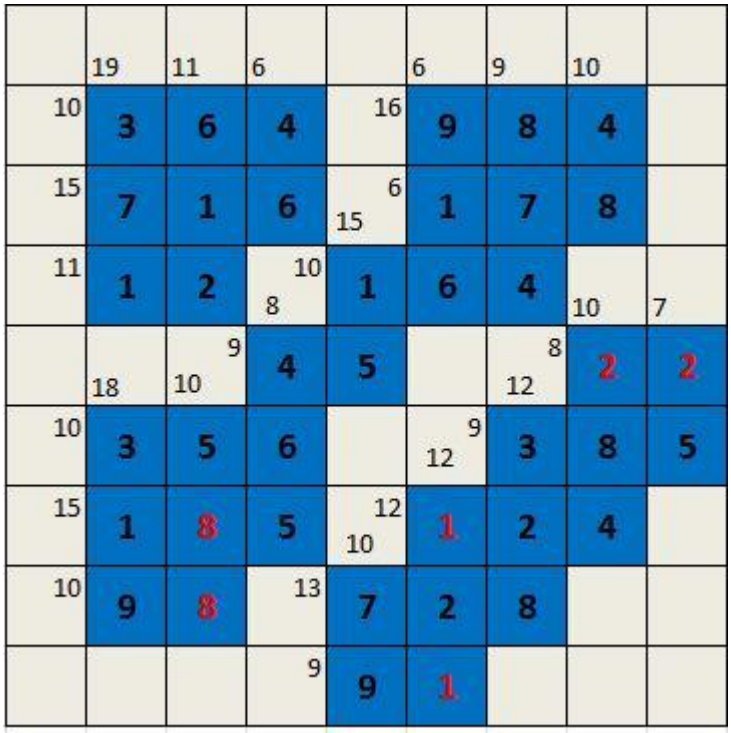
Gambar 3. 6 Penempatan Kromosom 1 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 1 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \qquad f_1 = \left(\frac{40 - 16}{40}\right) = 0{,}6$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}6$



	19	11	6		6	9	10	
10	3	6	4	16	9	8	4	
15	7	1	6	6 15	1	7	8	
11	1	2	10 8	1	6	4	10	7
	18	9 10	4	5		8 12	2	2
10	3	5	6		9 12	3	8	5
15	1	8	5	12 10	1	2	4	
10	9	8	13	7	2	8		
			9	9	1			

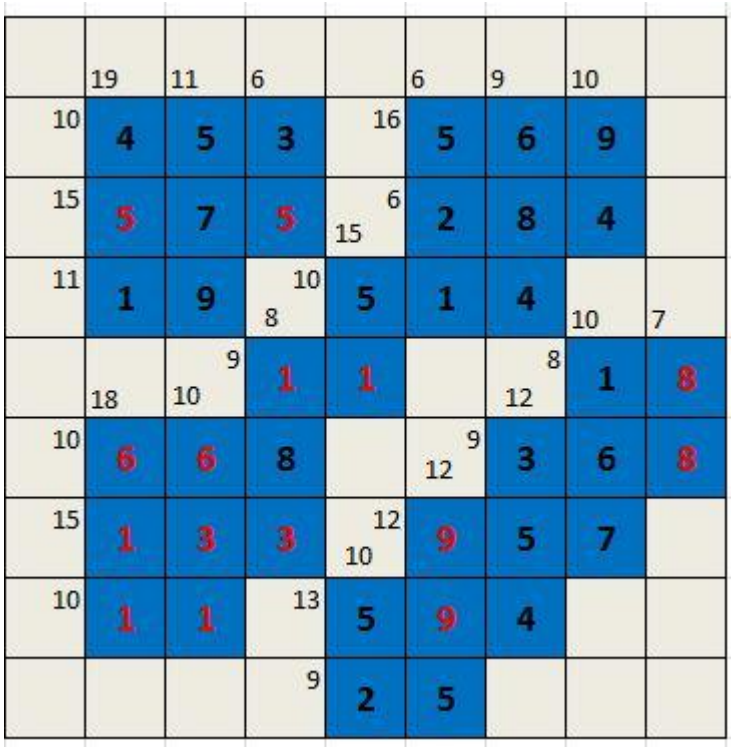
Gambar 3. 7 Penempatan Kromosom 2 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 2 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \qquad f_1 = \left(\frac{40 - 6}{40}\right) = 0{,}85$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}85$



	19	11	6		6	9	10	
10	4	5	3	16	5	6	9	
15	5	7	5	6 / 15	2	8	4	
11	1	9	10 / 8	5	1	4	10	7
	18	9 / 10	1	1		8 / 12	1	8
10	6	6	8		9 / 12	3	6	8
15	1	3	3	12 / 10	9	5	7	
10	1	1	13	5	9	4		
			9	2	5			

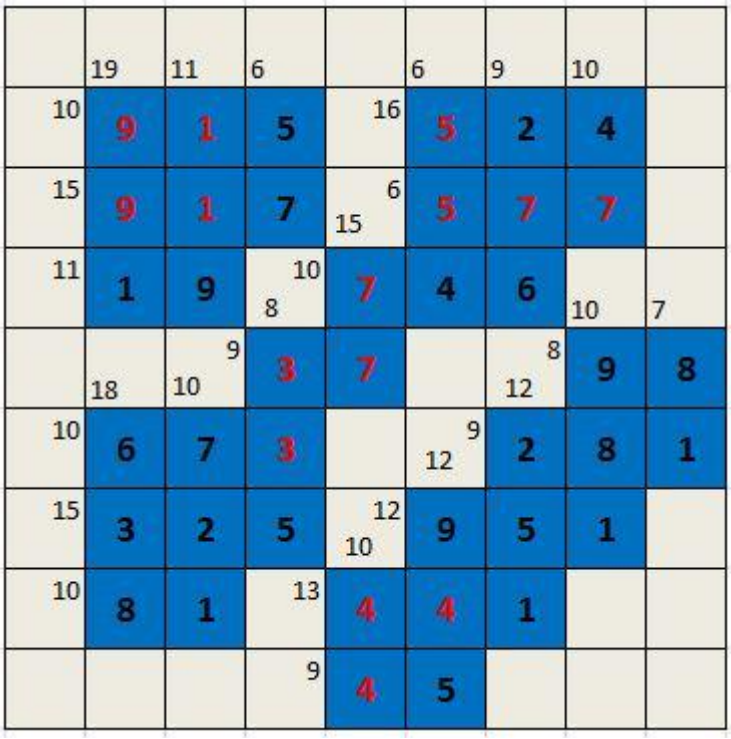
Gambar 3. 8 Penempatan Kromosom 3 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 3 :

$$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \qquad f_1 = \left(\frac{40 - 16}{40}\right) = 0{,}6$$

$$f_2 = \frac{LB}{JL} = 0$$

$$f_{total} = f_1 + f_2 = 0{,}6$$



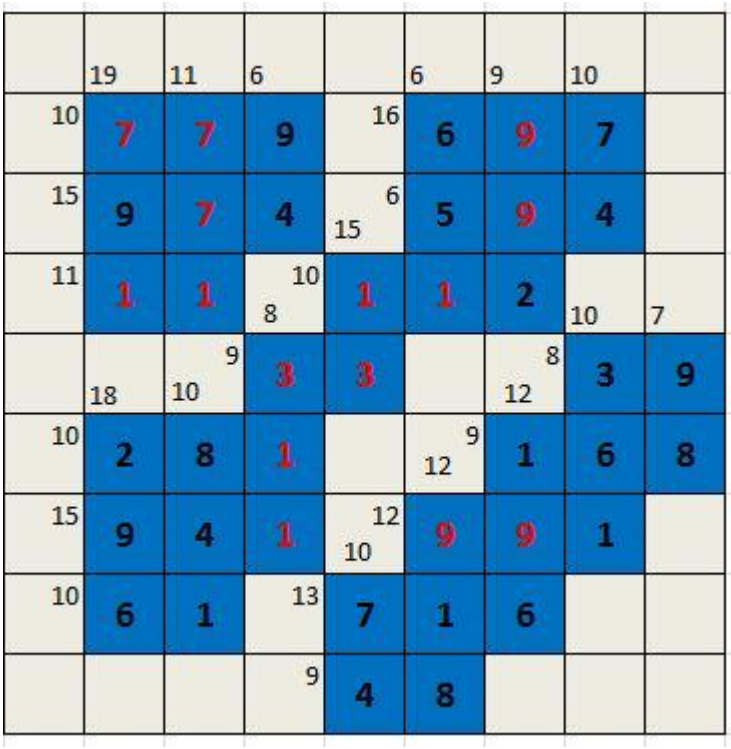
Gambar 3. 9 Penempatan Kromosom 4 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 4 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \quad f_1 = \left(\frac{40 - 16}{40}\right) = 0{,}6$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}6$



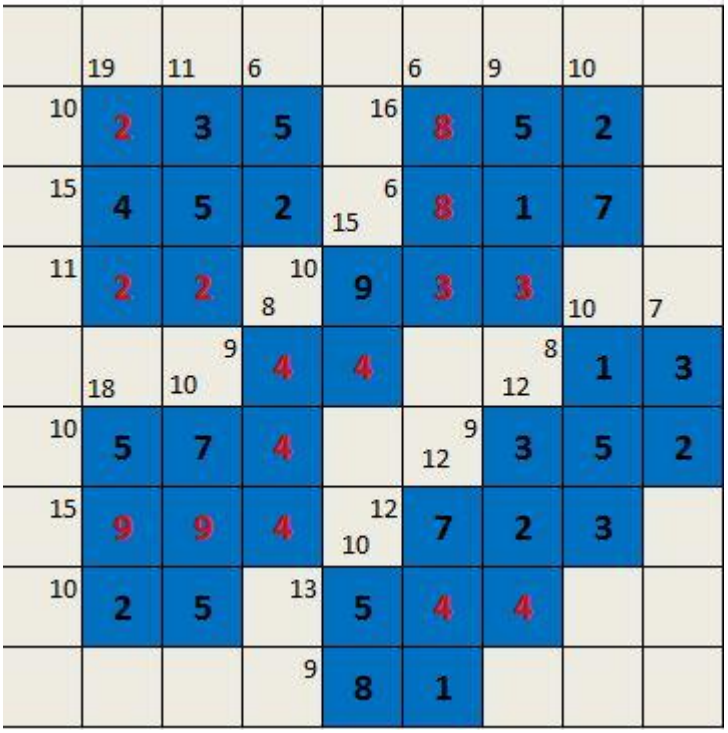
Gambar 3. 10 Penempatan Kromosom 5 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 5 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \quad f_1 = \left(\frac{40 - 16}{40}\right) = 0{,}6$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}6$



	19	11	6		6	9	10	
10	2	3	5	16	8	5	2	
15	4	5	2	6 15	8	1	7	
11	2	2	10 8	9	3	3	10	7
	18	9 10	4	4		8 12	1	3
10	5	7	4		9 12	3	5	2
15	9	9	4	12 10	7	2	3	
10	2	5	13	5	4	4		
			9	8	1			

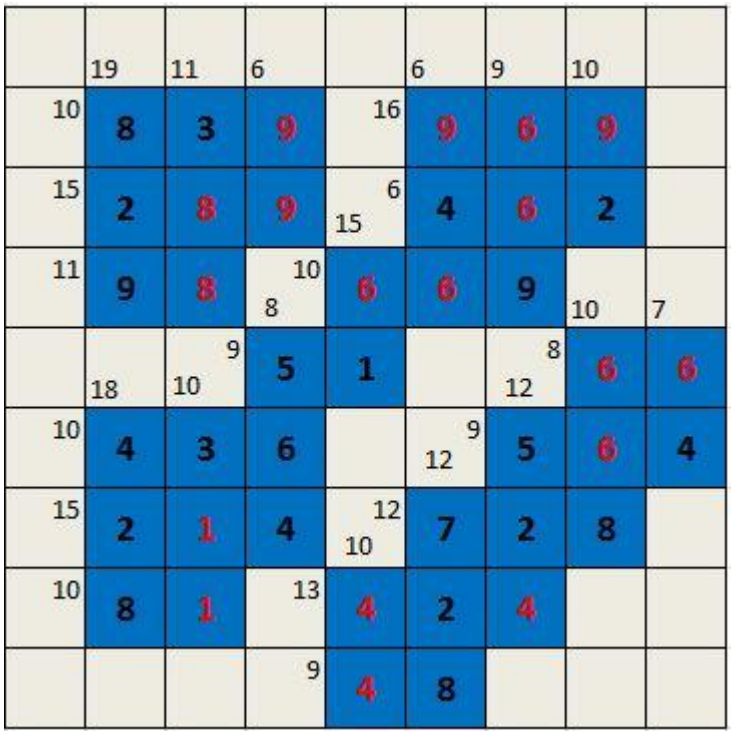
Gambar 3. 11 Penempatan Kromosom 6 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 6 :

$$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \qquad f_1 = \left(\frac{40 - 17}{40}\right) = 0{,}58$$

$$f_2 = \frac{LB}{JL} = 0$$

$$f_{total} = f_1 + f_2 = 0{,}58$$



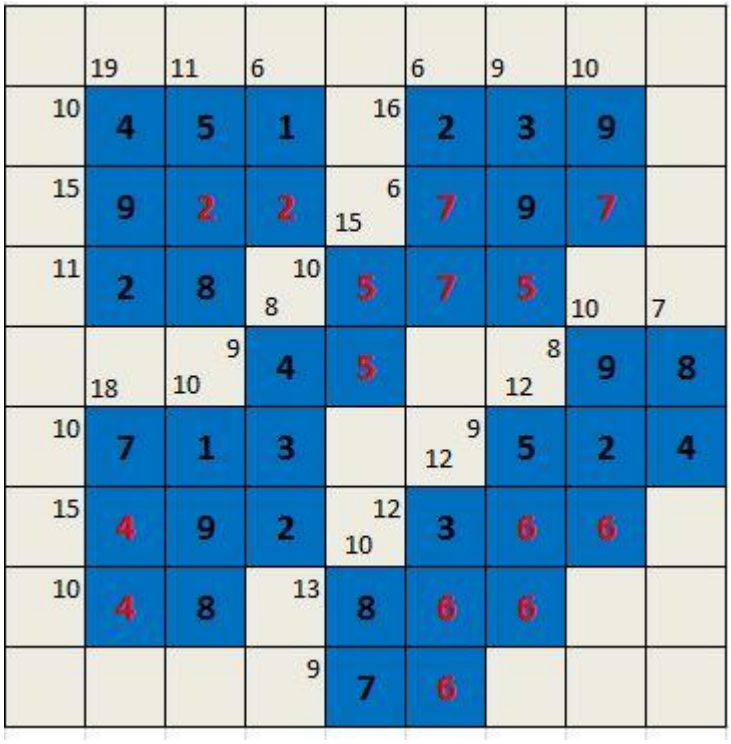
Gambar 3. 12 Penempatan Kromosom 7 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 7 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \quad f_1 = \left(\frac{40 - 20}{40}\right) = 0{,}5$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}5$



	19	11	6		6	9	10	
10	4	5	1	16	2	3	9	
15	9	2	2	6 15	7	9	7	
11	2	8	10 8	5	7	5	10	7
	18	9 10	4	5		8 12	9	8
10	7	1	3		9 12	5	2	4
15	4	9	2	12 10	3	6	6	
10	4	8	13	8	6	6		
			9	7	6			

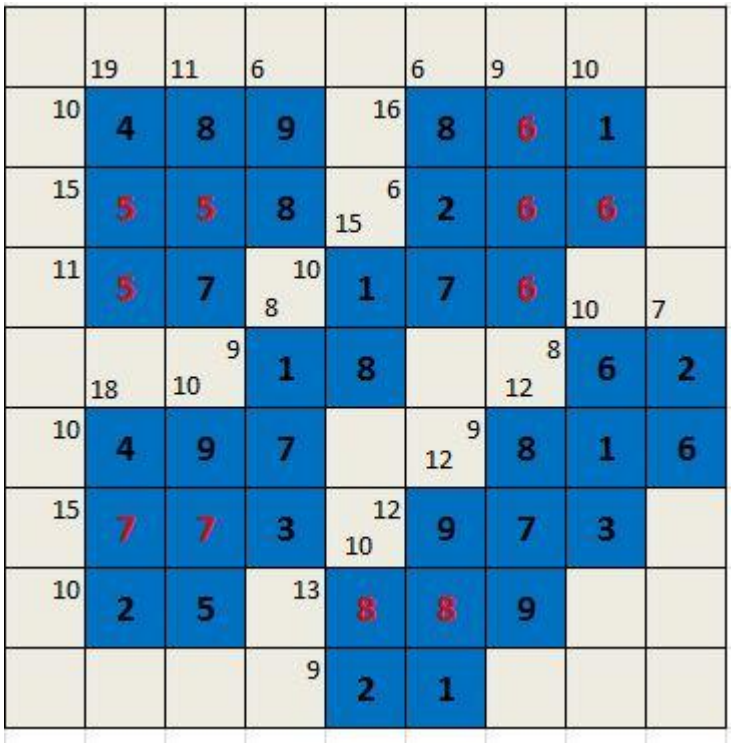
Gambar 3. 13 Penempatan Kromosom 8 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 8 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \quad f_1 = \left(\frac{40 - 20}{40}\right) = 0{,}5$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}5$



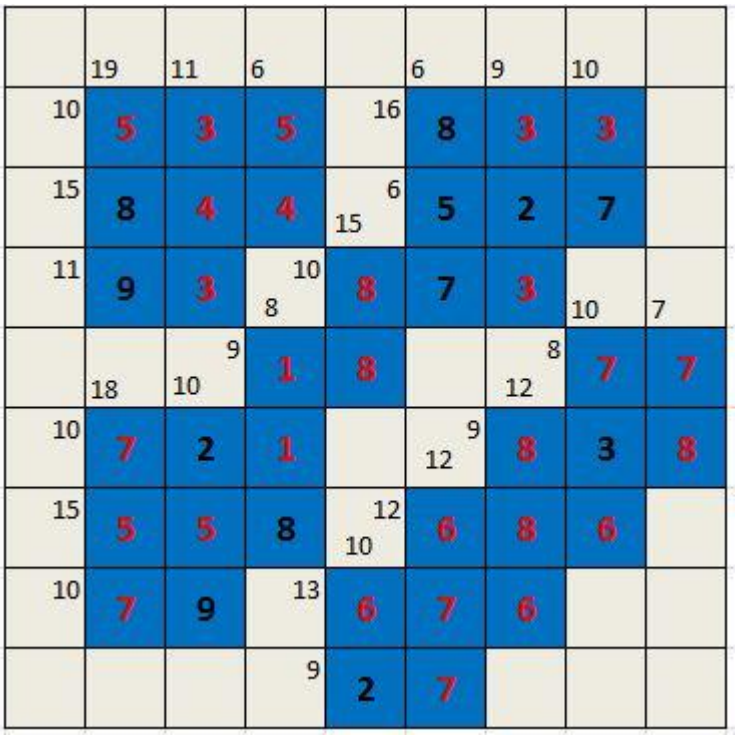
Gambar 3. 14 Penempatan Kromosom 9 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 9 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \qquad f_1 = \left(\frac{40 - 6}{40}\right) = 0{,}68$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}68$



Gambar 3. 15 Penempatan Kromosom 10 Pada *Puzzle* Kakuro

Perhitungan nilai *fitness* Kromosom 10 :

$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \qquad f_1 = \left(\frac{40 - 30}{40}\right) = 0{,}25$

$f_2 = \frac{LB}{JL} = 0$

$f_{total} = f_1 + f_2 = 0{,}25$

Berikut adalah hasil dari perhitungan nilai fitness dari seluruh kromosom awal yang dapat dilihat pada tabel 3.4 :

Tabel 3.4 Hasil Perhitungan Nilai *Fitness*

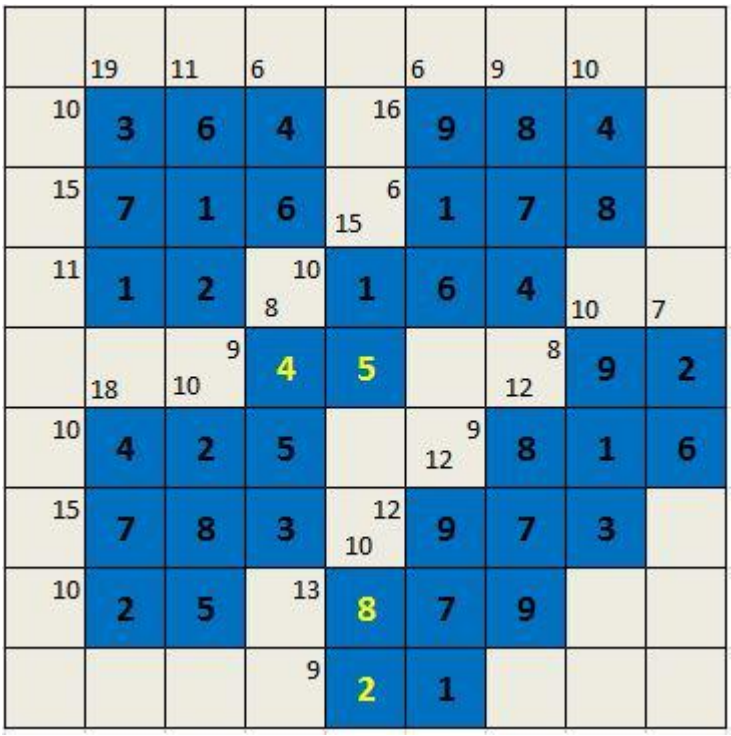
KROMOSOM	NILAI *FITNESS*
1	0,6
2	0,85
3	0,6

4	0,6
5	0,6
6	0,58
7	0,5
8	0,5
9	0,68
10	0,25

Dapat dilihat pada tabel 3.4 dari 10 kromosom pada generasi awal yang dilakukan sebanyak 40 generasi belum terdapat nilai *fitness* yang mencapai 2, kromosom dengan nilai lebih 1 baru terdapat pada generasi ke 32 yang dapat dilihat di lampiran D.

3.1.3.1.4 Proses Pengecekan Nilai Jawaban dan Nilai Soal

Proses pengecekan nilai jawaban dan nilai soal hanya dilakukan terhadap kromosom yang memiliki nilai f_1 sama dengan 1. Pada percobaan pertama yang dilakukan pada *puzzle* ukuran 9 x 9, nilai *fitness* 1 terdapat pada generasi ke 32 pada kromosom ke 17, pada lajur tersebut memiliki jumlah lajur yang benar sebanyak 2. Namun masih belum dinyatakan menyelesaikan permainan karena jumlah lajur yang harus benar adalah 30 lajur. Berikut adalah contoh proses pengecekan nilai soal pada kromosom yang dapat dilihat pada gambar 3.16 :



Gambar 3. 16 Proses Pengecekan Nilai Soal pada Kromosom

Perhitungan nilai *fitness* Kromosom 17 :

$$f_1 = \left(\frac{\mathrm{k} - \mathrm{e}}{k}\right), \qquad f_1 = \left(\frac{40 - 0}{40}\right) = 1$$

$$f_2 = \frac{LB}{JL}, \quad f_2 = \frac{2}{30} = 0{,}07$$

$$f_{total} = f_1 + f_2 = 1{,}07$$

Pada gambar 3.16 terdapat 2 lajur yang akumulasi nilai jawabannya sesuai dengan nilai soal.

3.1.3.1.5 Proses Rekombinasi (*Crossover*)

Proses rekombinasi atau c*rossover* merupakan proses menyilangkan dua kromosom induk atau *parents* untuk menghasilkan kromosom baru atau kromosom *offsprings* pada generasi selanjutnya. Pembentukan kromosom baru tersebut dengan memilih secara acak banyak posisi dalam kromosom induk yang dihasilkan pada proses seleksi dan kemudian saling menukar. Proses *crossover*

yang dilakukan pada penelitian ini menggunakan *crossover* satu titik serta probabilitas *crossover* (*Pc*) sebesar 0,6.

Jumlah *offspring* atau anak yang akan dihasilkan dari proses *crossover* ini dapat diperoleh dari perkalian nilai probabilitas *crossover* sebesar 0,6 dengan jumlah populasi sebesar 10, maka jumlah anak yang diperoleh adalah 6. Kromosom yang telah dilakukan proses seleksi masing-masing dibangkitkan bilangan acak antara 0 - 1 kemudian dibandingkan dengan *Pc*. Apabila nilai bilangan acak tersebut kurang atau sama dengan probabilitas *crossover* maka kromosom tersebut ditandai sebagai kromosom *parent.*

Untuk setiap proses crossover yang dilakukan ditetapkan akan menghasilkan 2 kromosom baru atau *offspring* sehingga akan dilakukan 3 proses *crossover*.

Berikut adalah hasil bilangan acak yang dibangkitkan pada setiap kromosom dapat dilihat pada tabel 3.5 :

Tabel 3.5 Hasil Bilangan Acak Tiap Kromosom

Kromosom	Nilai *Fitness*	Nilai Acak [0-1]	Status
1	0,6	0,93	-
2	0,85	0,05	*parent*
3	0,6	0,35	*parent*
4	0,6	0,08	*parent*
5	0,6	0,44	*parent*
6	0,58	0,84	-
7	0,5	0,68	-
8	0,5	0,17	*parent*
9	0,68	0,71	-
10	0,25	0,24	*parent*

Berdasarkan hasil dari tabel 3.5 maka kromosom yang akan menjadi *parent* pada proses *crossover* adalah kromosom 2, kromosom 3, kromosom 4, kromosom

5, kromosom 8 dan kromosom 10. Pada contoh *puzzle* dengan ukuran 9 x 9 titik potong yang dipilih pada proses crossover ditentukan secara statis yaitu pada titik ke 21, sedangakan titik potong pada *puzzle* dengan ukuran 14 x 14 yaitu pada titik ke 52. Berikut adalah gambaran dari proses crossover berdasarkan dari kromosom yang telah ditetapkan menjadi parent :

1. Crossover Kromosom 10 dan Kromosom 3

   Kromosom 10 :

   5 3 5 8 3 3 8 4 4 5 2 7 9 3 8 7 3 1 8 7 7 7 2 1 8 3 8 5 5 8 6 8 6 7 9 6 7 6 2 7

   Kromosom 3 :

   4 5 3 5 6 9 5 7 5 2 8 4 1 9 5 1 4 1 1 1 8 6 6 8 3 6 8 1 3 3 9 5 7 1 1 5 9 4 2 5

   Hasil Crossover 1 (Offspring 1) :

   5 3 5 8 3 3 8 4 4 5 2 7 9 3 8 7 3 1 8 7 7 6 6 8 3 6 8 1 3 3 9 5 7 1 1 5 9 4 2 5

   Hasil Crossover 2 (Offspring 2) :

   4 5 3 5 6 9 5 7 5 2 8 4 1 9 5 1 4 1 1 1 8 7 2 1 8 3 8 5 5 8 6 8 6 7 9 6 7 6 2 7

2. Crossover Kromosom 2 dan Kromosom 4

   Kromosom 2 :

   3 6 4 9 8 4 7 1 6 1 7 8 1 2 1 6 4 4 5 2 2 3 5 6 3 8 5 1 8 5 1 2 4 9 8 7 2 8 9 1

   Kromosom 4 :

   9 1 5 5 2 4 9 1 7 5 7 7 1 9 7 4 6 3 7 9 8 6 7 3 2 8 1 3 2 5 9 5 1 8 1 4 4 1 4 5

   Hasil Crossover 3 (Offspring 3) :

   3 6 4 9 8 4 7 1 6 1 7 8 1 2 1 6 4 4 5 2 2 6 7 3 2 8 1 3 2 5 9 5 1 8 1 4 4 1 4 5

   Hasil Crossover 4 (Offspring 4) :

   9 1 5 5 2 4 9 1 7 5 7 7 1 9 7 4 6 3 7 9 8 3 5 6 3 8 5 1 8 5 1 2 4 9 8 7 2 8 9 1

3. Crossover Kromosom 5 dan Kromosom 8

   Kromosom 5 :

   7 7 9 6 9 7 9 7 4 5 9 4 1 1 1 1 2 3 3 3 9 2 8 1 1 6 8 9 4 1 9 9 1 6 1 7 1 6 4 8

   Kromosom 8 :

   4 5 1 2 3 9 9 2 2 7 9 7 2 8 5 7 5 4 5 9 8 7 1 3 5 2 4 4 9 2 3 6 6 4 8 8 6 6 7 6

   Hasil Crossover 5 (Offspring 5) :

   7 7 9 6 9 7 9 7 4 5 9 4 1 1 1 1 2 3 3 3 9 7 1 3 5 2 4 4 9 2 3 6 6 4 8 8 6 6 7 6

   Hasil Crossover 6 (Offspring 6) :

   4 5 1 2 3 9 9 2 2 7 9 7 2 8 5 7 5 4 5 9 8 2 8 1 1 6 8 9 4 1 9 9 1 6 1 7 1 6 4 8

Berikut adalah hasil offspring dari proses crossover yang dapat dilihat pada tabel 3.6 :

Tabel 3.6 Hasil *Offspring* Dari Proses *Crossover*

Offspring	Kromosom
1	5 3 5 8 3 3 8 4 4 5 2 7 9 3 8 7 3 1 8 7 7 6 6 8 3 6 8 1 3 3 9 5 7 1 1 5 9 4 2 5
2	4 5 3 5 6 9 5 7 5 2 8 4 1 9 5 1 4 1 1 1 8 7 2 1 8 3 8 5 5 8 6 8 6 7 9 6 7 6 2 7
3	3 6 4 9 8 4 7 1 6 1 7 8 1 2 1 6 4 4 5 2 2 6 7 3 2 8 1 3 2 5 9 5 1 8 1 4 4 1 4 5
4	9 1 5 5 2 4 9 1 7 5 7 7 1 9 7 4 6 3 7 9 8 3 5 6 3 8 5 1 8 5 1 2 4 9 8 7 2 8 9 1
5	7 7 9 6 9 7 9 7 4 5 9 4 1 1 1 1 2 3 3 3 9 7 1 3 5 2 4 4 9 2 3 6 6 4 8 8 6 6 7 6
6	4 5 1 2 3 9 9 2 2 7 9 7 2 8 5 7 5 4 5 9 8 2 8 1 1 6 8 9 4 1 9 9 1 6 1 7 1 6 4 8

3.1.3.1.6 Proses Mutasi

Proses selanjutnya yang akan dilakukan adalah proses mutasi. Mutasi akan dilakukan adalah *swap mutation* sehingga akan ada 2 nilai yang akan ditukar pada

satu kromosom yang letaknya ditentukan secara acak [**3**]. Nilai probabilitas mutasi (Pm) yang digunakan pada penelitian ini adalah 0,001. Untuk menentukan jumlah *offspring* yang akan dihasilkan dari proses mutasi adalah dengan mengalikai nilai Pm 0,001 dengan jumlah populasi sebesar 10, sehingga diperoleh jumlah *offspring* dengan jumlah 1.

Untuk memilih kromosom yang akan dimutasi maka perlu dibangkitkan suatu bilangan acak. Kromosom yang telah dilakukan proses seleksi masing-masing dibangkitkan bilangan acak antara 0 - 1 kemudian dibandingkan dengan *Pm*. Apabila nilai bilangan acak tersebut kurang atau sama dengan probabilitas mutasi maka kromosom tersebut akan dimutasi. Kromosom yang akan dimutasi adalah kromosom 9. Titik dari nilai gen yang akan ditukar adalah titik ke 7 dan 24.

Berikut adalah proses mutasi yang dilakukan pada kromosom 9 :

Kromosom 9 :

4 8 9 8 6 1 5 5 8 2 6 6 5 7 1 7 6 1 8 6 2 4 9 7 8 1 6 7 7 3 9 7 3 2 5 8 8 9 2 1

Hasil Mutasi (*Offspring* 7) :

4 8 9 8 6 1 7 5 8 2 6 6 5 7 1 7 6 1 8 6 2 4 9 5 8 1 6 7 7 3 9 7 3 2 5 8 8 9 2 1

3.1.3.1.7 Proses Seleksi

Proses seleksi yang akan dilakukan pada penelitian ini menggunakan metode *ranking*. Nilai *fitness* yang didapat dari proses evaluasi diurutkan dengan cara kromosom terbaik diberi nilai indeks *fitness* baru sebesar 1, kromosom terbaik kedua diberi nilai indeks fitness baru sebesar 2, dan seterusnya. Kromosom terburuk diberi nilai indeks fitness baru sebesar n dimana n adalah banyak kromosom dalam suatu populasi.

Berikut adalah hasil dari proses seleksi yang dilakukan pada seluruh kromosom yang terdapat pada tabel 3.7 :

Tabel 3.7 Hasil Proses Seleksi *Rank*

Kromosom	Nilai *Fitness*	*Rank*
1	0,6	8
2	0,85	17
3	0,6	9
4	0,6	10
5	0,6	11
6	0,58	7
7	0,5	5
8	0,5	6
9	0,68	14
10	0,25	1
11	0,35	2
12	0,45	4
13	0,85	16
14	0,65	13
15	0,4	3
16	0,6	12
17	0,78	15

Hasil dari proses seleksi akan digunakan sebagai acuan untuk membentuk populasi baru yang akan digunkan pada proses berikutnya. Hanya kromosom dengan *rank* 8 sampai 17 saja yang akan digunakan untuk generasi berikutnya.

3.1.3.1.8 Pembentukan Populasi Baru

Berdasarkan hasil dari proses seleksi, terdapat 17 kromosom dan hanya 10 populasi dengan nilai *fitness* terbaik yang akan bertahan.

Berikut adalah hasil dari proses pembentukan populasi baru yang dapat dilihat pada tabel 3.8 :

Tabel 3.8 Hasil Proses Pembentukan Populasi Baru

Kromosom	Nilai *Fitness*	Seleksi Rank	Status
1	0,6	8	Bertahan
2	0,85	17	Bertahan
3	0,6	9	Bertahan
4	0,6	10	Bertahan
5	0,6	11	Bertahan
6	0,58	7	Dihapus
7	0,5	5	Dihapus
8	0,5	6	Dihapus
9	0,68	14	Bertahan
10	0,25	1	Dihapus
11	0,35	2	Dihapus
12	0,45	4	Dihapus
13	0,85	16	Bertahan
14	0,65	13	Bertahan
15	0,4	3	Dihapus
16	0,6	12	Bertahan
17	0,78	15	Bertahan

Berikut adalah daftar 10 kromosom baru yang akan digunakan pada generasi berikutnya yang terdapat pada tabel 3.9 :

Tabel 3.9 Daftar Kromosom Yang Akan Digunakan Untuk Generasi Berikutnya

Kromosom	Nilai *Fitness*
1	0,6
2	0,6
3	0,6
4	0,6
5	0,6

6	0,65
7	0,68
8	0,78
9	0,85
10	0,85

3.1.3.1.9 Hasil Proses Algoritma Genetika

Hasil dari algoritma genetika yang diharapkan untuk menyelesaikan permainan *puzzle* kakuro adalah kromosom dengan nilai f_{total} sama dengan 2. Hasil dari percobaan yang dilakukan pada *puzzle* kakuro ukuran 9 x 9, adalah sebagai berikut :

1. Pada generasi awal nilai fitness terbaik adalah 0,85.
2. Nilai f_1 1 pada percobaan pertama baru didapatkan pada generasi ke 32 dan hanya 2 dari 30 lajur saja yang akumulasi nilai jawabannya sama dengan nilai soal.
3. Pada percobaan kedua dilakukan selama 10 menit, menghasilkan 198.517 generasi, hasilnya 18 dari 30 lajur yang akumulasi nilai jawabannya sama dengan nilai soal.
4. Pada percobaan ketiga dilakukan selama 30 menit menghasilkan 606.319 generasi, hasilnya 18 dari 30 lajur yang akumulasi nilai jawabannya sama dengan nilai soal.

Dari percobaan – percobaan yang telah dilakukan dapat ditarik kesimpulan bahwa algoritma genetika tidak dapat digunakan untuk menyelesaikan *puzzle* kakuro, dikarenakan kromosom yang dihasilkan hanya memenuhi aturan tidak ada nilai yang sama pada setiap lajur tetapi tidak memenuhi aturan nilai jawaban setiap lajur harus sama dengan nilai soalnya. Untuk data hasil percobaan dapat dilihat di lampiran D.

3.1.4 Analisis Kebutuhan Sistem

Spesifikasi kebutuhan sistem didapat berdasarkan kebutuhan perangkat lunak untuk mengimplementasikan algoritma genetika sebagai *generator* dan

solver permainan *puzzle* kakuro. Analisis kebutuhan perangkat lunak terbagi menjadi dua bagian, yaitu :

1. Kebutuhan Non Fungsional
2. Kebutuhan Fungsional

3.1.4.1 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional adalah langkah dimana seorang pembangun perangkat lunak (*software developer*) menganalisis sumber daya yang akan digunakan dan menggunakan perangkat lunak yang dibangun. Perangkat keras dan perangkat lunak yang dimiliki harus sesuai dengan kebutuhan atau permintaan, sehingga dapat ditentukan kompabilitas aplikasi yang dibangun terhadap sumber daya yang ada.

Analisis kebutuhan non fungsional yang dilakukan dibagi dalam tiga tahap, yaitu:

1. Analisis Perangkat Keras
2. Analisis Perangkat Lunak
3. Analisis Pengguna

3.1.4.1.1 Analisis Perangkat Keras

Spesifikasi perangkat keras minimum yang butuhkan untuk menjalankan aplikasi pengimplementasian algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro, dapat dilihat pada tabel 3.10 :

Tabel 3.10 Spesifikasi Perangkat Keras

No	Perangkat Keras	Spesifikasi
1	Prosesor	Kecepatan 1,3 Ghz
2	Monitor	Monitor 14”
3	RAM	2048 MB

3.1.4.1.2 Analisis Perangkat Lunak

Spesifikasi perangkat lunak yang dibutuhkan untuk menjalankan aplikasi pengimplementasian algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro, dapat dilihat pada tabel 3.11 :

Tabel 3.11 Spesifikasi Perangkat Lunak

No	Perangkat Lunak
1	Sistem Operasi Windows 7
2	Visual Studio 2010

3.1.4.1.3 Analisis Kebutuhan Perangkat Pikir

Analisis kebutuhan perangkat pikir merupakan uraian mengenai siapa saja yang akan menggunakan sistem dan terlibat dalam pengolahan data beserta karakteristiknya sehingga dapat diketahui tingkat pengalaman dan pemahaman pengguna terhadap sistem. Adapun karakteristik dari pengguna dapat dilihat pada tabel 3.12 :

Tabel 3.12 Spesifikasi Pengguna

No	Pengguna	Tanggung Jawab	Keterampilan
1	*User*	a. Melakukan pembangkitan nilai soal *puzzle*. b. Menyelesaikan permainan *puzzle* kakuro dengan penerapan algoritma genetika.	a. Menguasai dalam penggunaan komputer. b. Mengetahui aturan permainan *puzzle* kakuro

3.1.4.2 Analisis Kebutuhan Fungsional

Spesifikasi kebutuhan fungsional adalah spesifikasi tentang hal-hal yang akandilakukan sistem ketika diimplementasikan. Analisis kebutuhan ini diperlukan untuk menentukan keluaran yang akan dihasilkan sistem, masukan

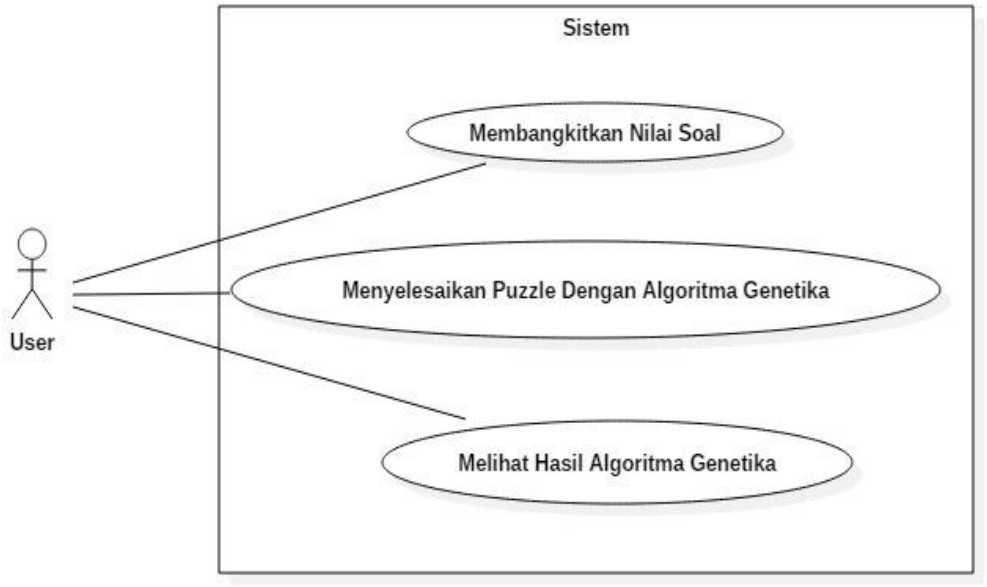
yang diperlukan sistem, lingkup proses yang digunakan untuk mengolah masukan menjadi keluaran.

3.1.4.2.1 Use Case Diagram

Use case merupakan gambaran umum dari perancangan sistem yang akan dibangun. Terdapat tiga poin yang merincikan use case diagram ini, yaitu :

1. Identifikasi Use Case.
2. Identifikasi Aktor.
3. Identifikasi Skenario Use case.

Adapun use case diagram dari implementasi algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro dapat dilihat pada gambar 3.17 :



Gambar 3.17 *Use Case* Diagram Implementasi Algoritma Genetika Untuk Menyelesaikan Permainan *Puzzle* Kakuro

3.1.4.2.1.1 Identifikasi Use Case Diagram

Identifikasi *Use Case* berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use Case Diagram*. Definisi *Use Case* diterangkan pada tabel 3.13 :

Tabel 3.13 Identifikasi Use Case

No	*Use Case*	Deskripsi
1	Membangkitkan Nilai Soal	Fungsionalitas untuk membangkitkan nilai soal *puzzle* kakuro.
2	Menyelesaikan Permainan Dengan Algoritma Genetika	Fungsionalitas untuk menyelesaikan permainan *puzzle* kakuro dengan menggunakan penerapan dari algoritma genetika.
3	Melihat Hasil Algoritma Genetika	Fungsionalitas untuk melihat hasil dari algoritma genetika.

3.1.4.2.1.2 Identifikasi Aktor

Identifikasi *Actor* berfungsi untuk menjelaskan *Actor* yang terdapat pada *Use Case Diagram*. Identifikasi *Actor* diterangkan pada tabel 3.14 :

Tabel 3.14 Identifikasi Actor

No	*Actor*	Deskripsi
1	User	Orang yang menggunakan sistem

3.1.4.2.1.3 Identifikasi Skenario Use Case Diagram

Skenario setiap bagian pada *use case* menunjukkan proses apa yang terjadi pada setiap bagian didalam use case tersebut, dimana user memberikan perintah pada setiap bagian dan respon apa yang diberikan oleh sistem kepada user setelah user memberikan perintah pada setiap bagian-bagian *use case*.

Skenario use case untuk membangkitkan nilai soal dapat dilihat pada tabel 3.15.

Tabel 3.15 Skenario Use Case Membangkitkan Nilai Soal

Aktor *(actor)*	User
Nama use case *(use case name)*	Membangkitkan Nilai Soal
Deskripsi *(goal in context)*	Membangkitkan nilai soal pada

	permainan *puzzle* kakuro.
Kondisi awal *(precondition)*	Menampilkan *form Puzzle* Kakuro.
Kondisi sukses *(succesfull end condition)*	Berhasil membangkitkan nilai soal dari *puzzle* kakuro.
Kondisi gagal *(failed end condition)*	-
Include case *(include case)*	-
Skenario Normal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan *form Puzzle* Kakuro
2. Memilih proses pembangkitan nilai jawaban dan nilai soal.	
	3. Melakukan proses pembangkitan nilai jawaban secara acak hingga memenuhi aturan permainan.
	4. Mengakumulasikan nilai jawaban untuk menjadi nilai soal setiap lajur *puzzle*.
	5. Menampilkan pesan nilai soal berhasil dibangkitkan dan menampilkan hasil nilai jawaban dan nilai soal.

Skenario use case untuk menyelesaikan permainan dengan menggunakan algoritma genetika dapat dilihat pada tabel 3.16.

Tabel 3.16 Skenario Use Case Menyelesaikan Permainan Dengan Algoritma Genetika

Aktor *(actor)*	User
Nama use case *(use case name)*	Menyelesaikan Permainan Dengan Algoritma Genetika
Deskripsi *(goal in context)*	Menyelesaikan permainan *puzzle* kakuro

	dengan menggunakan penerapan algoritma genetika
Kondisi awal *(precondition)*	Nilai soal telah dibangkitkan.
Kondisi sukses *(succesfull end condition)*	*Puzzle* berhasil diselesaikan.
Kondisi gagal *(failed end condition)*	*Puzzle* gagal untuk diselesaikan.
Include case *(include case)*	-
Skenario Normal	
Aksi Aktor	**Reaksi Sistem**
1. Mengisi waktu percobaan.	
2. Memilih proses penyelesaian *puzzle*.	3. Membentuk populasi awal secara acak
	4. Menghitung nilai *fitness* dari setiap kromosom.
	5. Melakukan pengecekan apakah terdapat kromosom dengan nilai *fitness* 1.
	6. Jika terdapat kromosom dengan nilai *fitness* 1 maka akan dilakukan pengecekan apakah akumulasi nilai jawaban sama dengan nilai yang ada pada soal permainan. Jika seluruh nilai soal sesuai dengan akumulasi nilai jawaban maka akan keluar pesan bahwa *puzzle* kakuro telah diselesaikan.
	7. Jika tidak maka akan masuk ke proses *crossover*.
	8. Membangkitkan bilangan acak untuk tiap kromosom sebagai acuan untuk proses *crossover*.

	9. Jika nilai acak kromosom lebih kecil atau sama dengan 0,6 maka kromosom tersebut akan menjadi parent untuk proses *crossover*.
	10. Melakukan proses *crossover*
	11. Menghasilkan kromosom *offspring*
	12. Menghitung nilai *fitness* dari setiap kromosom *offspring*.
	13. Membangkitkan bilangan acak untuk tiap kromosom sebagai acuan untuk proses mutasi.
	14. Jika nilai acak kromosom lebih kecil atau sama dengan 0,001 maka kromosom tersebut akan menjadi parent untuk proses mutasi.
	15. Melakukan proses mutasi
	16. Menghasilkan *offspring*
	17. Melakukan proses seleksi dan pembentukan populasi baru.
	18. Kromosom yang bertahan akan kembali ke proses evaluasi(menghitung nilai *fitness*).

Skenario use case untuk melihat hasil algoritma genetika dapat dilihat pada tabel 3.17.

Tabel 3.17 Skenario Use Case Melihat Hasil Algoritma Genetika

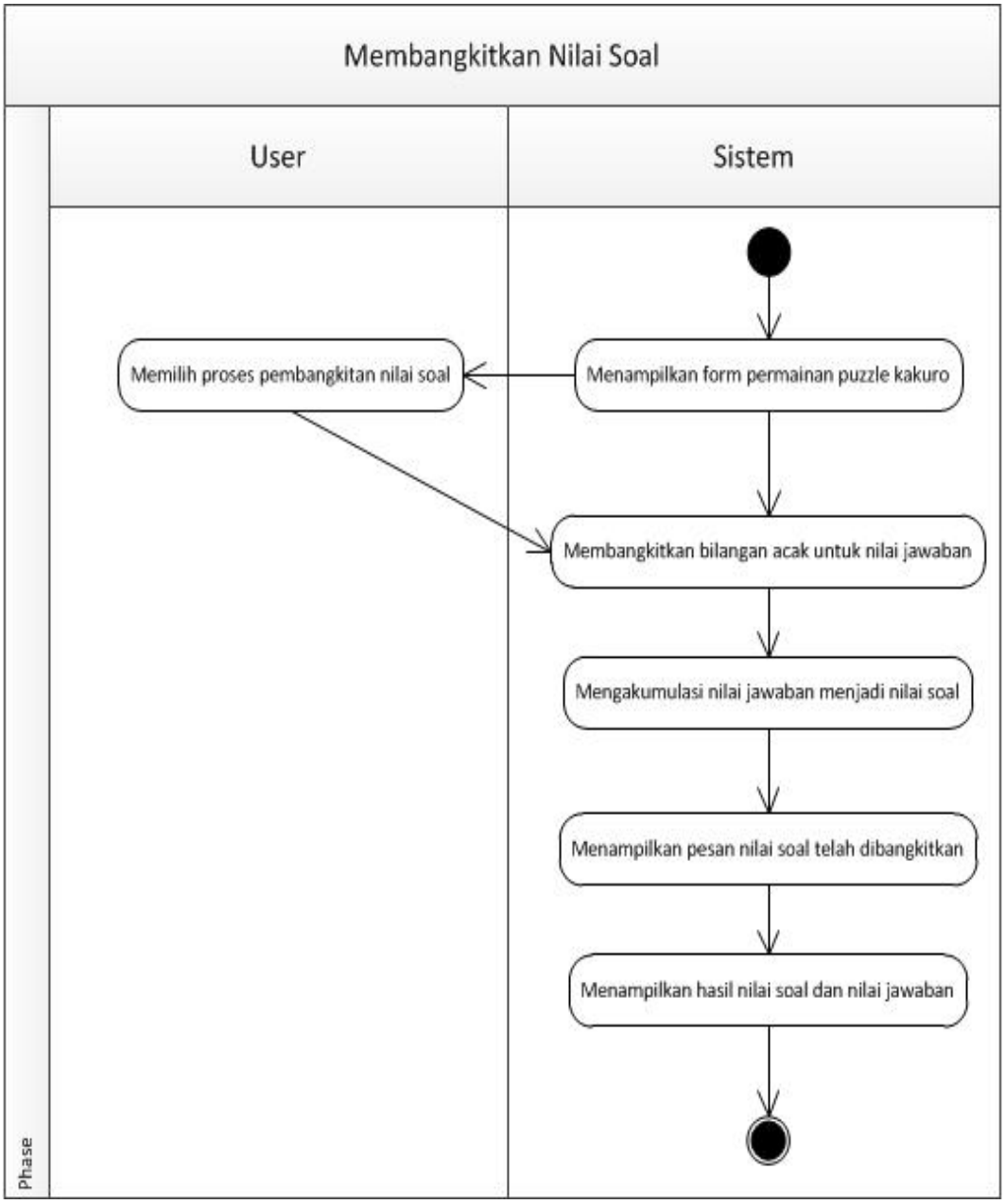
Aktor *(actor)*	User
Nama use case *(use case name)*	Melihat Hasil Algoritma Genetika
Deskripsi *(goal in context)*	Melihat hasil proses algoritma genetika

	setiap generasinya.
Kondisi awal *(precondition)*	Nilai soal telah dibankitkan
Kondisi sukses *(succesfull end condition)*	Menampilkan hasil algoritma genetika.
Kondisi gagal *(failed end condition)*	-
Include case *(include case)*	-
Skenario Normal	
Aksi Aktor	**Reaksi Sistem**
1. Mengisi waktu percobaan.	
2. Memilih pilihan untuk menampilkan hasil algoritma genetika.	
3. . Memilih proses penyelesaian *puzzle*.	4. Melakukan proses penyelesaian permainan *puzzle* kakuro.
	5. Menampilkan hasil algoritma genetika.

3.1.4.2.2 Activity Diagram

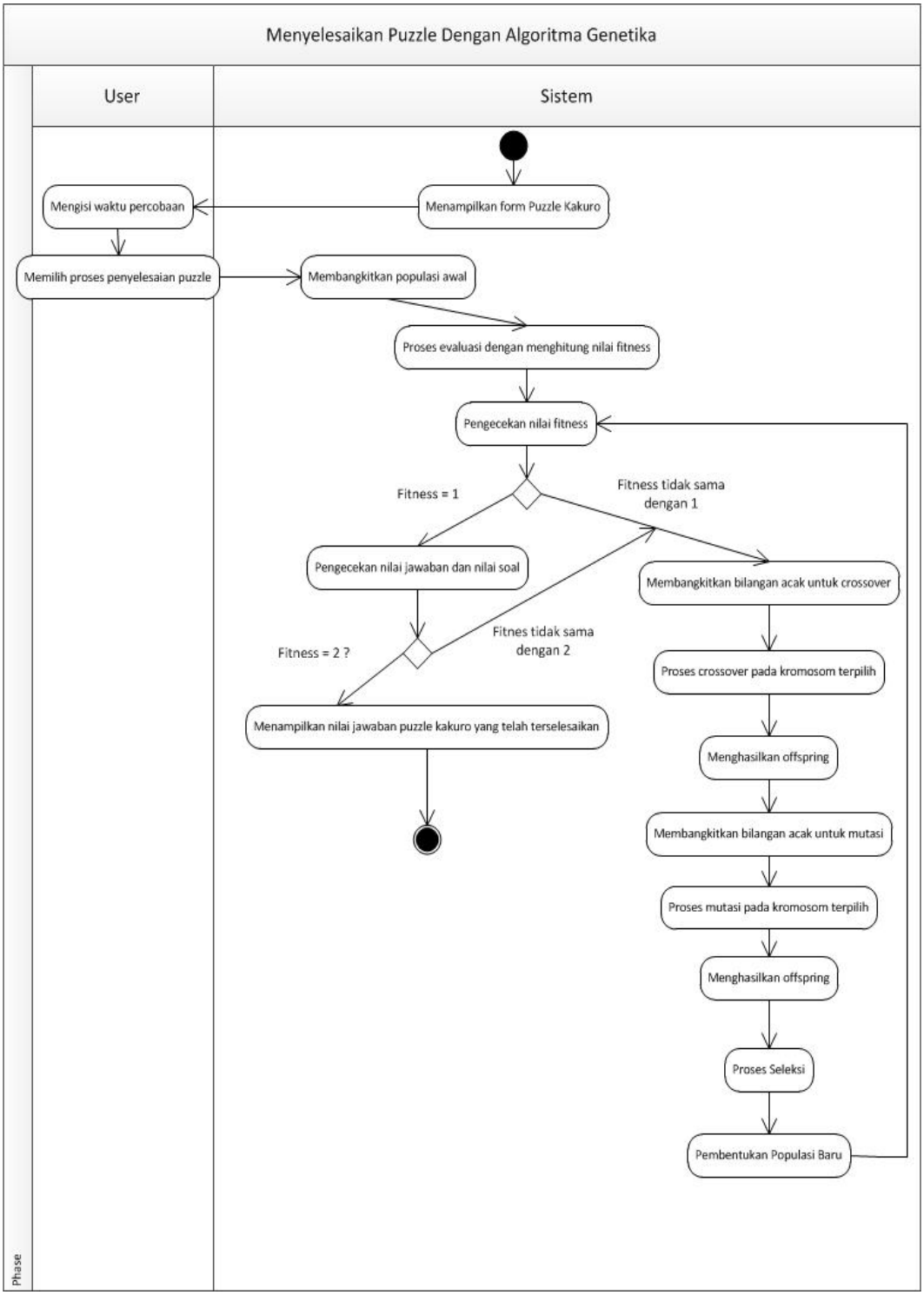
Diagram aktivitas (*Activity Diagram*) memodelkan alur kerja (*workflow*) sebuah proses bisnis dan urutan aktivitas dalam suatu proses. Berikut adalah *Activity diagram* untuk implementasi algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro :

Untuk diagram aktivitas membangkitkan nilai soal dapat dilihat pada gambar 3.18.



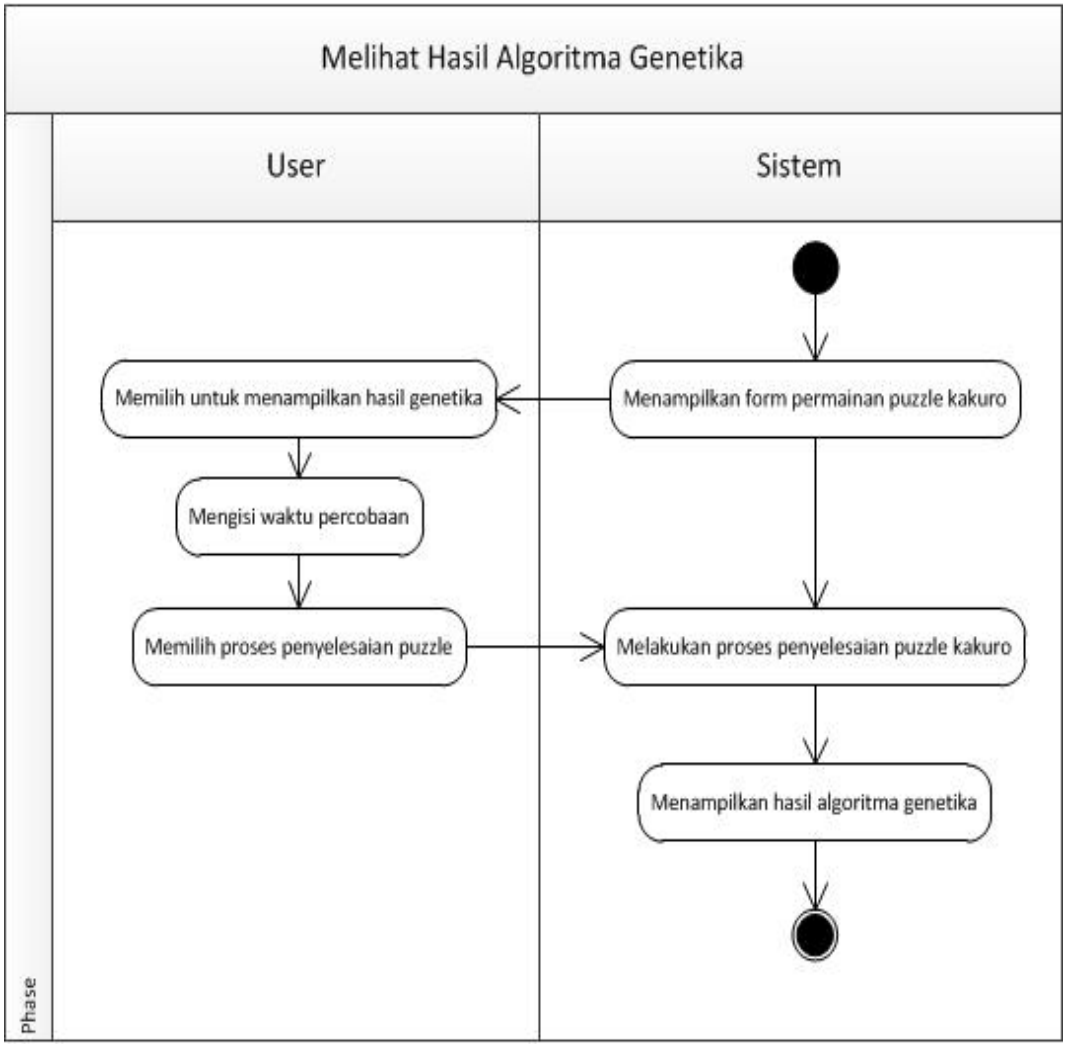
Gambar 3. 18 Diagram *Activity* Membangkitkan Nilai Soal

Untuk diagram aktivitas dalam melakukan proses menyelesaikan *puzzle* kakuro dengan algoritma genetika dilihat pada gambar 3.19.



Gambar 3.19 Diagram *Activity* Menyelesaikan *Puzzle* Dengan Algoritma Genetika

Untuk diagram aktivitas lihat hasil algoritma genetika dapat dilihat pada gambar 3.20.

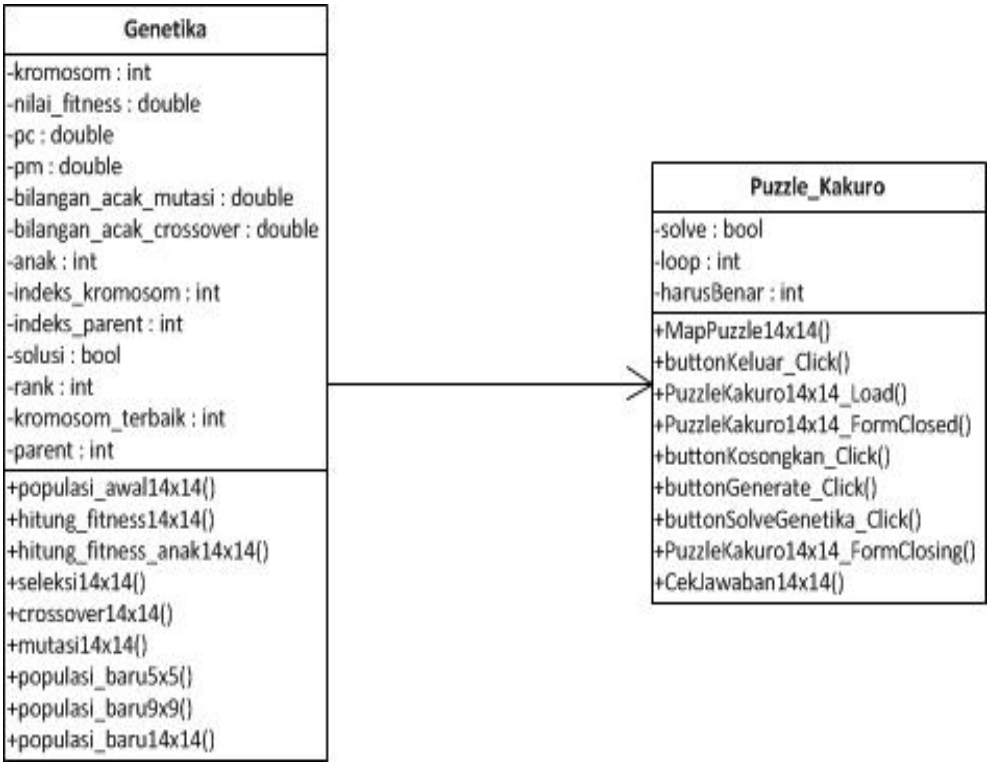


Gambar 3. 20 Diagram *Activity* Melihat Hasil Algortima Genetika

3.1.4.2.3 Class Diagram

Class diagram membantu menggambarkan struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling sering ditemui dalam pemodelan sistem berbasis *object-oriented.*

Perancangan kelas diagram dari implementasi algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro dapat dilihat pada gambar 3.21.



Gambar 3.21 Diagram Kelas Implementasi Algoritma Genetika Untuk Menyelesaikan Permainan *Puzzle* Kakuro

3.1.4.2.4 Sequence Diagram

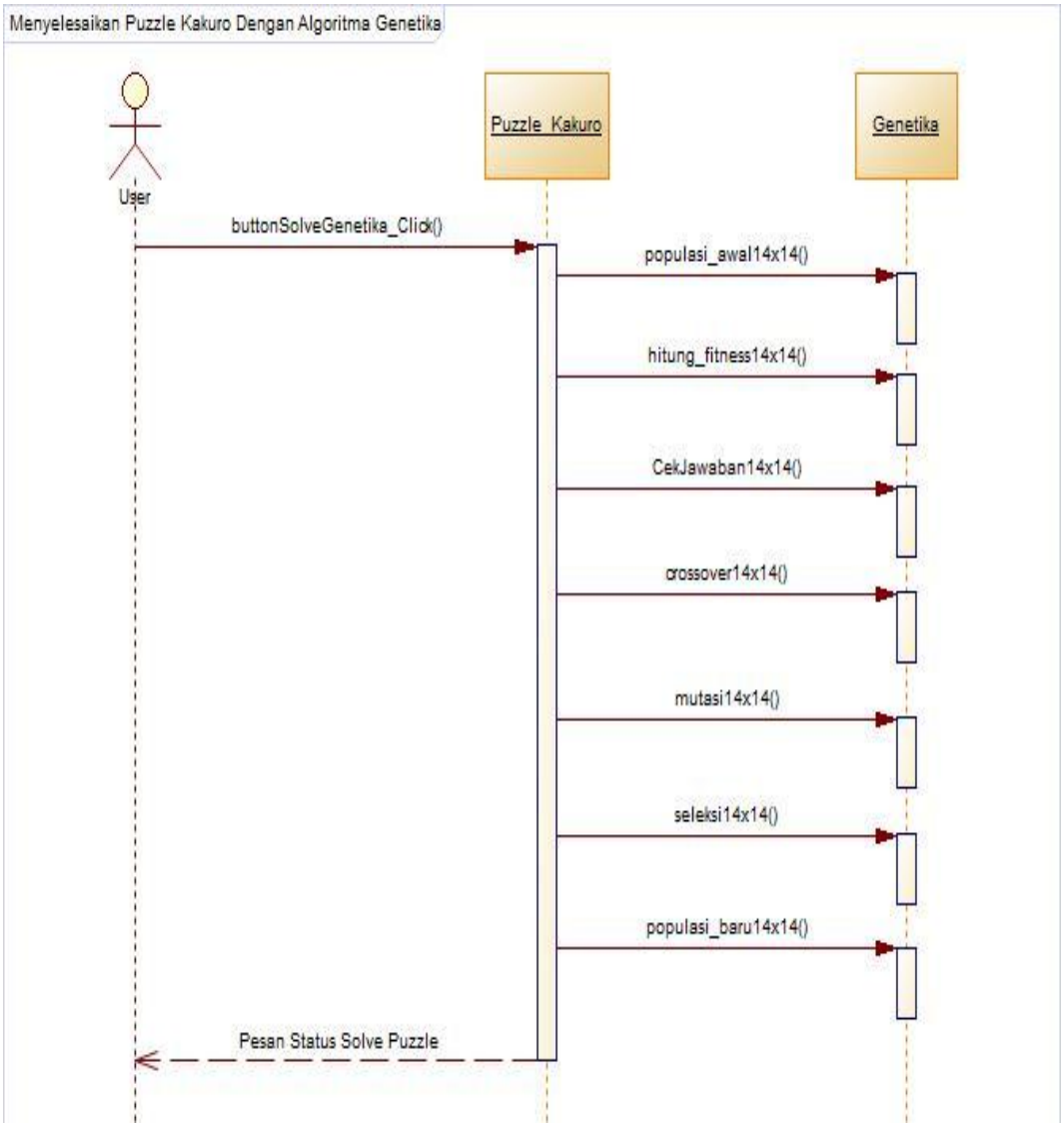
Sequence diagram menggambarkan interaksi antara sejumlah objek dalam urutan waktu. Adapun rincian dari *sequence diagram* implementasi algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro adalah sebagai berikut:

Untuk diagram sekuen dari proses pembangkitan nilai soal dapat dilihat pada gambar 3.22.



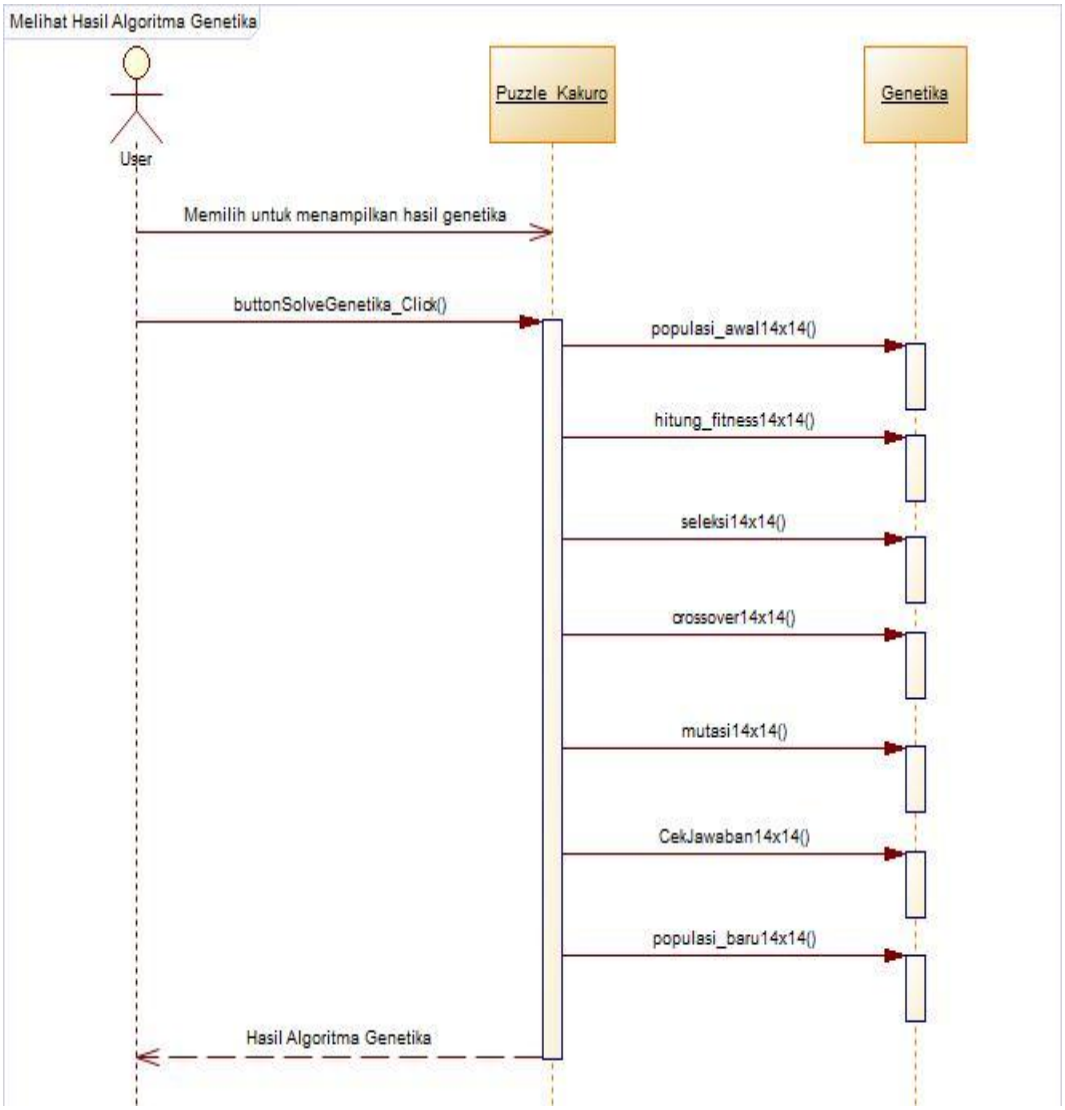
Gambar 3. 22 Diagram Sekuen Membangkitkan Nilai Soal

Untuk diagram sekuen dari proses penyelesaian permainan *puzzle* kakuro dapat dilihat pada gambar 3.23.



Gambar 3. 23 Diagram Sekuen Menyelesaikan *Puzzle* Kakuro Dengan Algoritma Genetika

Diagram sekuen untuk melihat hasil algoritma genetika dapat dilihat pada gambar 3.24.



Gambar 3. 24 Diagram Sekuen Melihat Hasil Algoritma Genetika

3.2 Perancangan Sistem

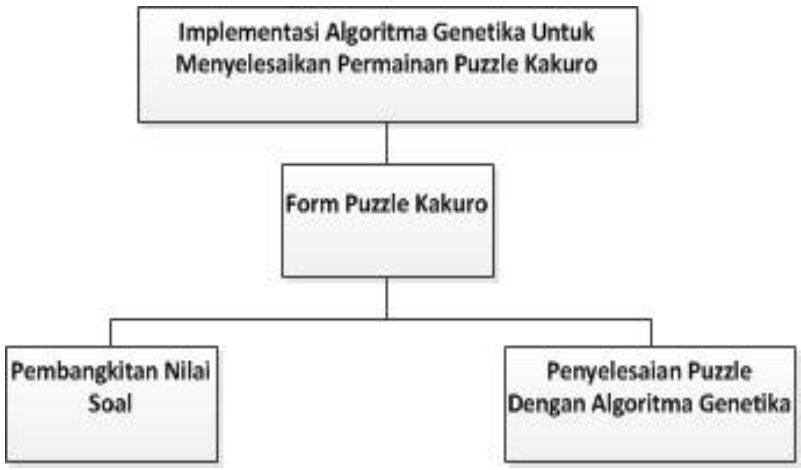
Perancangan sistem merupakan suatu proses yang menggambarkan bagaimana suatu sistem dibangun untuk memenuhi kebutuhan pada fase analisis. Tahap perancangan sistem terdiri dari perancangan arsitektur.

3.2.1 Perancangan Arsitektur

Perancangan arsitektur terdiri dari perancangan struktur menu, perancangan antarmuka, perancangan pesan dan perancangan jaringan semantik.

3.2.1.1 Perancangan Struktur Menu

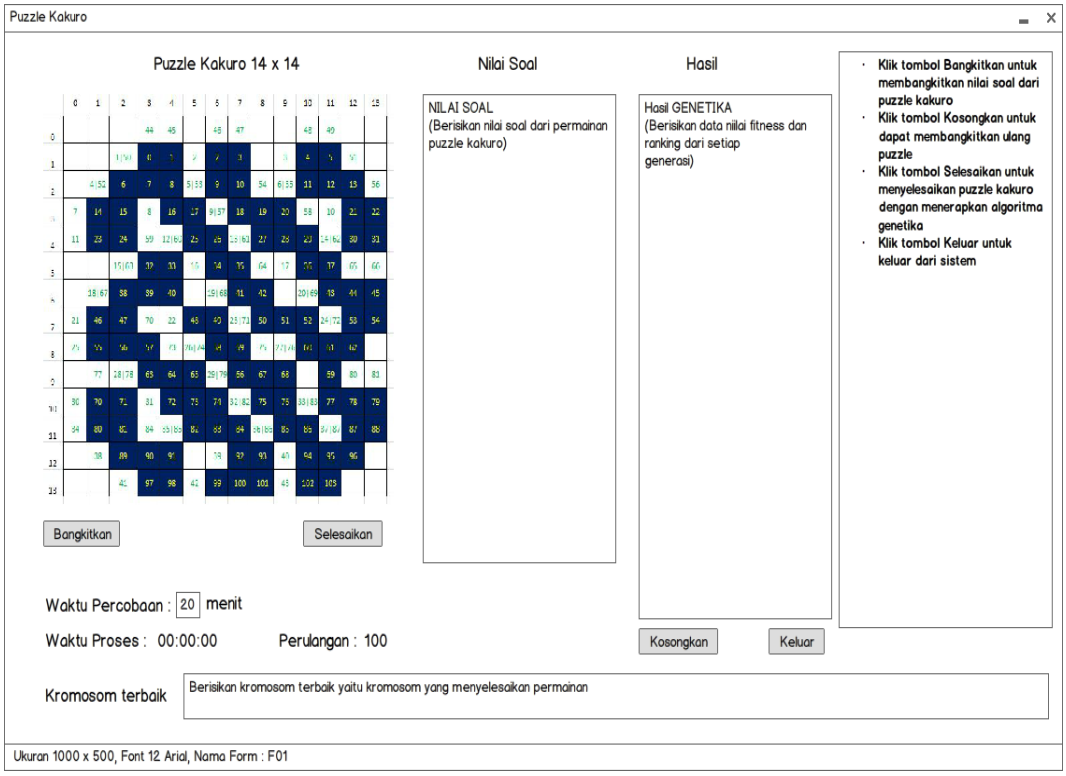
Perancangan menu dilakukan untuk mempermudah interaksi antara sistem dengan pengguna. Perancangan struktur menu pada aplikasi ini dapat dilihat pada gambar 3.25 :



Gambar 3.25 Perancangan Struktur Menu

3.2.1.2 Perancangan Antarmuka

Tahap perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi. Adapun perancangan antarmuka dari implementasi algoritma genetika untuk menyelesaikan permainan *puzzle* kakuro dapat dilihat pada gambar 3.26.

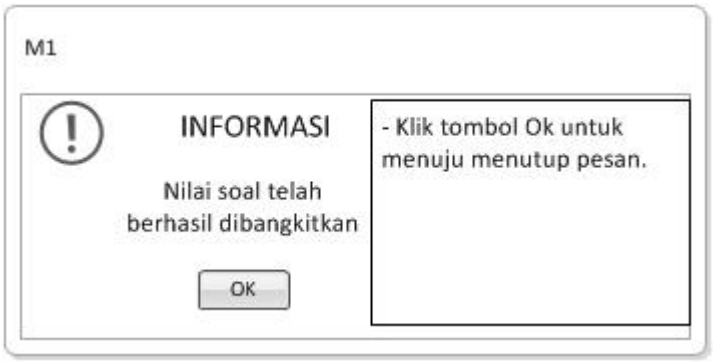


Gambar 3.26 Perancangan Antarmuka

3.2.1.3 Perancangan Pesan

Pesan merupakan tampilan dari suatu perangkat lunak yang berfungsi menyampaikan notifikasi dan informasi kepada pengguna agar perangkat lunak lebih interaktif. Terdapat beberapa perancangan pesan pada pembangunan aplikasi untuk mengimplementasikan algoritma pada *puzzle* kakuro ini, diantaranya :

1. Pesan Informasi Nilai Soal Berhasil Diangkitkan



Gambar 3.27 Perancangan Pesan Informasi Nilai Soal Berhasil Dibangkitkan

2. Pesan Informasi *Puzzle* Berhasil Diselesaikan



Gambar 3.28 Perancangan Pesan Informasi *Puzzle* Berhasil Diselesaikan

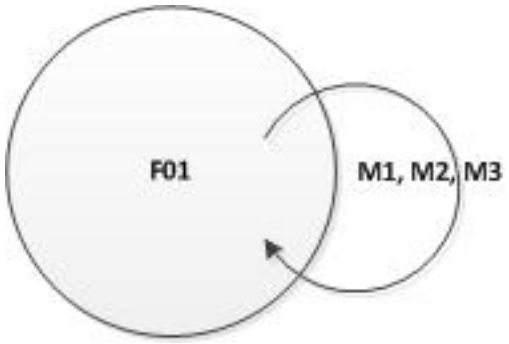
3. Pesan Konfirmasi Keluar dari Sistem



Gambar 3.29 Perancangan Pesan Konfirmasi Untuk Keluar dari Sistem

3.2.1.4 Jaringan Semantik

Jaringan semantik adalah diagram yang menggambarkan aliran-aliran menu dan pesan dalam sebuah program. Adapun jaringan semantik dari aplikasi yang akan dibangun dapat dilihat pada gambar 3.30 :



Gambar 3. 30 Jaringan Semantik dari Implementasi Algoritma Genetika Untuk Menyelesaikan Permainan *Puzzle* Kakuro