PROBLEM SOLVING SUDOKU MENGGUNAKAN ALGORITMA GENETIKA

JIMMY HADINATA

Sekolah Tinggi Manajemen Informatika dan Komputer Pontianak
Program Studi Teknik Informatika
Jln. Merdeka N0. 372 Pontianak, Kalimantan Barat
E-mail : **vern.mell@yahoo.com**

***Abstracts**: Sudoku is a logical puzzle that using numbers as the symbol to play, it has a quite interesting problem to be solved, Genetic Algorithm is suited as a problem solving for sudoku because Genetic algorithm method is to find solution with an optimization when Sudoku need an optimization to find the solution. Designing Sudoku with Genetic Algorithm as the problem solver could solve sudoku faster because Genetic Algorithm always produce a new better solution every loop that called Generations with selection, crossover and mutation process to create a new solution.*

***Keywords**: Sudoku, Genetic Algorithm, Problem Solving, Generations, Selection, Crossover, Mutation*

1. PENDAHULUAN

Teknologi komputer saat ini sudah berkembang sangat cepat, baik dari sisi hardware ataupun dari sisi software. Dari sisi Software, berbagai aplikasi kini dapat diciptakan dengan mudahnya, karena perkembangan berbagai jenis software seiring dengan perkembangan Kecerdasan Buatan.

Kecerdasan Buatan banyak digunakan dalam aplikasi-aplikasi sebagai fitur tambahan. teknologi komputer yang didukung kecerdasan buatan sekarang ini sudah dapat mengatasi berbagai masalah yang ada pada saat perkembangan komputer sampai pada sekarang ini.

Banyaknya aplikasi-aplikasi yang diciptakan dengan kecerdasan buatan didalamnya, terutama aplikasi-aplikasi game pada dunia komputer, seperti aplikasi game real strategi, game sport, game yang bersifat petualangan (*adventure*) dan juga game yang bersifat mengasah logika.

Salah satu game logika yang kini mulai banyak diminati oleh banyak kalangan adalah "Sudoku". Sudoku adalah sebuah permainan teka-teki angka sederhana yang telah dikenal seluruh dunia mulai dari anak-anak sampai orang tua. Di Jepang, ribuan *puzzle* mampu diselesaikan setiap harinya. Bahkan di beberapa sekolah sudoku telah menjadi bagian dari kurikulum. Untuk memainkan *puzzle* angka ini pemain tidak perlu berpikir keras, tidak harus mempunyai intelektualitas yang tinggi, dan tidak perlu handal berhitung, karena yang dibutuhkan hanya kemampuan berpikir secara logika, kesabaran tinggi dan ketajaman akurasi.

Merancang Sudoku yang interaktif diperlukan solusi algoritma yang untuk mengacak (*generate*) dan menemukan solusi pada sudoku karena *puzzle* ini menggunakan prinsip keunikan dan menghasilkan banyak kombinasi angka yang membuat permainan sudoku dengan banyak kemungkinan tercipta, untuk menghadapi permasalahan tersebut, digunakan Algoritma Genetika.

Algoritma Genetika merupakan Kercerdasan Buatan yang menggunakan seleksi alamiah dan genetika alamiah dimana yang lebih kuat yang akan bertahan, sehingga

kemungkinan-kemungkinan yang dianggap lemah akan dieliminasi disetiap reproduksinya untuk mendapatkan hasil terbaik dan memperpendek waktu pencarian karena ruang lingkup diperkecil setiap reproduksinya.

Aplikasi permainan sudoku yang didukung dengan kecerdasan buatan yang berupa Algoritma Genetika yang tidak memutasi gen yang dianggap baik sehingga dapat mencari solusi dengan *looping* yang lebih sedikit dan waktu yang lebih cepat untuk menghasilkan solusi

2. TINJAUAN PUSTAKA

2.1 Algoritma Genetika

Menurut Thiang (2001), Algoritma Genetika adalah suatu algoritma pencarian yang bersifat stokastik, berbasis pada mekanisme seleksi alam dan genetika. Menurut Suyanto (2007:205) "Algoritma Genetika adalah algoritma pencarian yang didasarkan pada mekanisme seleksi alamiah dan genetika alamiah.".

Menurut Suyanto (2007), Istilah seleksi alamiah secara sederhana diilustrasikan dalam kasus populasi jerapah. Pada suatu kondisi dimana jumlah makanan sangat terbatas dan berada di tempat yang tinggi, maka jerapah berleher panjang akan bertahan hidup karena bisa menjangkau dedaunan yang berada di tempat tinggi, sedangkan jerapah berleher pendek akan mati. Dalam hal ini, jerapah berleher panjang di katakan sebagai individu yang memiliki kualitas tinggi sehingga lolos dari proses seleksi alamiah.

Menurut Suyanto (2007), Genetika alamiah adalah mekanisme yang sangat rumit. Sampai saat ini, ilmu pengetahuan hanya dapat menjelaskan hal tersebut secara sederhana sebagai berikut. Dalam ilmu biologi, sekumpulan individu yang sama hidup, bereproduksi, dan mati dalam suatu area yang disebut populasi. Jika anggota-anggota populasi (individu) terpisah, misalkan karena bencana alam, maka individu-individu tersebut membentuk beberapa populasi yang terpisah. Melanie Mitchel (1999:8) menyatakan, "Secara sederhana algoritma genetika memiliki 3 operator, yaitu seleksi, pindah silang, dan mutasi."

Secara sederhana, Algoritma Genetika bekerja seperti berikut (Melanie Mitchel, 1999:8-9):

1. Mulai dengan mengacak populasi dari kromosom *n 1-bit* (Kandidat solusi dari masalah).
2. Hitung *fitness f(x)* untuk setiap kromosom dalam populasi.
3. Ulangi langkah berikut sampai offspring dibentuk:
   a. Pilih sepasang kromosom dari populasi, kemungkinan terpilih dilihat dari besar fitnessnya.
   b. Lakukan *crossover* secara acak untuk menghasilkan 2 offspring.
   c. Mutasikan offspring untuk menghasilkan kromosom pada populasi baru.
2. Gantikan populasi sekarang dengan populasi baru
3. Ulangi langkah 2

2.2 Skema Pengkodean

Untuk dapat di proses menggunakan Algoritma Genetika, suatu permasalahan harus dikonversi dulu ke dalam bentuk individu yang diwakili oleh satu atau lebih kromosom dengan kode tertentu. Algoritma Genetika merepresentasikan gen secara umum sebagai bilangan *real*, *decimal* atau *biner*, yaitu (Suyanto, 2007:209):

1. *Real-number encoding*. Pada skema ini, nilai gen berada dalam interval [0,R], dimana R adalah bilangan real positif dan biasanya R=1.

Kromosom 1	3,0
Kromosom 2	5,0

Gambar 1. Contoh *Real-number Encoding*

2. *Descrete decimal encoding*. Pada skema ini, setiap gen bisa berupa deretan bilangan bulat dalam interval [0,9].

Kromosom 1	478513269
Kromosom 2	438517269

Gambar 2. Contoh *Descrete Decimal Encoding*

3. *Binary encoding*. Setiap gen bisa berupa deretan nilai 0 atau 1.

Kromosom 1	101111101010
Kromosom 2	001111111111

Gambar 3. Contoh *Binary Encoding*

2.3 Nilai Fitness

Menurut Edi Satriyanto (2009:72), Nilai *fitness* adalah nilai yang menyatakan baik tidaknya suatu solusi. Nilai *fitness* ini yang dijadikan acuan dalam mencapai nilai optimal dalam algoritma genetika. Jika solusi yang dicari adalah memaksimalkan sebuah fungsi h (dikenal sebagai masalah maksimasi), maka nilai *fitness* yang digunakan adalah nilai dari fungsi h tersebut, yakni $f = h$ (dimana f adalah nilai *fitness*). Tetapi jika masalahnya adalah meminimalkan fungsi h (masalah minimasi), maka fungsi h tidak bisa digunakan secara langsung. Hal ini disebabkan Algoritma Genetika menggunakan suatu aturan bahwa individu yang memiliki nilai *fitness* lebih tinggi akan memiliki kemampuan bertahan hidup lebih tinggi dari pada individu yang bernilai fitness rendah. Oleh karena itu, nilai fitness untuk masalah minimasi adalah $f = 1/h$, yang artinya semakin kecil nilai h semakin besar nilai f.

2.4 Seleksi Orang Tua

Seleksi digunakan untuk memilih individu-individu mana saja yang akan dipilih untuk proses *crossover* atau mutasi. Seleksi digunakan untuk mendapatkan calon induk yang baik. Semakin tunggi nilai fitness suatu individu semakin besar kemungkinannya untuk terpilih.

Ada beberapa metode seleksi, antara lain(Sri Kusumadewi,2003:284-289) : a) *Rank-based fitness assignment,* populasi diurutkan menurut nilai objektifnya. Nilai fitness dari tiap-tiap individu hanya tergantung pada posisi individu tersebut dalam urutan; b) *Rouletee wheel selection,* Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran fitnessnya. Sebuah bilangan random dibangkitkan dan individu yang memiliki segmen dalam kawasan bilangan random tersebut akan terseleksi; c) *Stochastic universal sampling,* Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran fitnessnya seperti halnya pada seleksi roulette. Kemudian diberikan sejumlah pointer sebanyak individu

yang ingin diseleksi pada garis tersebut; d) *Local selection,* pada seleksi lokal, setiap individu yang berada di dalam konstrain tertentu disebut dengan nama lingkungan lokal. Interaksi antar individu hanya dilakukan didalam wilayah tersebut; e) *Truncation selection,* pada seleksi pemotongan ini, lebih terkesan sebagai seleksi buatan. Pada metode ini, individu-individu diurutkan berdasarkan nilai fitnessnya; dan f) *Tournament selection, p*ada metode seleksi dengan turnamen ini, akan ditetapkan suatu nilai tour untuk individu-individu yang dipilih secara random dari suatu populasi.

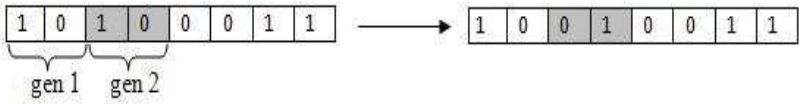
2.5 Mutasi

Mutasi berperan untuk menggantikan informasi bit yang hilang akibat proses seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada insialisasi populasi. Berdasarkan bagian yang termutasi, proses mutasi dapat dibedakan atas tiga bagian (Suyanto,2007:213-214):

1) Mutasi pada tingkat kromosom : semua gen dalam kromosom berubah.

Gambar 4. Contoh Mutasi gen tingkat kromosom

2) Mutasi pada tingkat gen : semua bit dalam satu gen akan berubah



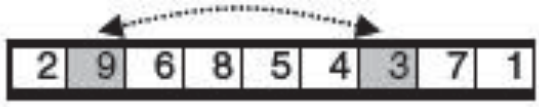
Gambar 5. Contoh Mutasi pada tingkat gen

3) Mutasi pada tingkat : hanya 1 bit yang berubah.

Gambar 6. Contoh Mutasi pada tingkat

Berdasarkan mutasinya dalam gen, terbagi menjadi 3, yaitu (Mantere dan Koljonen,2006:88):

1) *Swap Mutation* : nilai dari 2 posisi ditukar

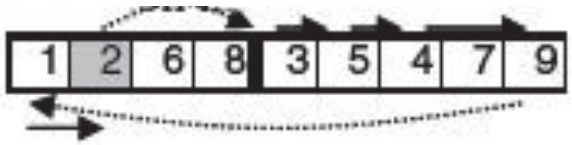


Gambar 7. Contoh *Swap Mutation*

2) *3 Swap Mutation* : nilai dari 3 posisi diputar, baik searah jarum jam ataupun berlawanan

Gambar 8. Contoh *3 Swap Mutation*

3) *Insertion Mutation* : 1 atau lebih nilai dimasukkan pada posisi tertentu dan nilai yang lain ikut berputar searah jarum jam ataupun berlawanan.



Gambar 9. Contoh *Insertion Mutation*

2.6 Elitism

Menurut Suyanto (2007), Seleksi yang dilakukan secara acak, menyebabkan tidak ada jaminan bahwa suatu individu bernilai *fitness* tertinggi akan selalu terpilih. Kalaupun individu bernilai *fitness* tertinggi terpilih, mungkin saja individu tersebut akan rusak karena proses pindah silang. Untuk menjaga agar individu bernilai *fitness* tertinggi tersebut tidak hilang selama evolusi, perlu dibuat satu atau dua kopinya. Prosedur ini dikenal sebagai elitisme. Prosedur ini hanya digunakan pada Algoritma Genetika berjenis *generational replacement*.

2.7 Penggantian Populasi

Menurut Suyanto (2007), Pengantian populasi untuk Algoritma memiliki beberapa prosedur yang dapat digunakan, diantaranya adalah selalu mengganti individu yang memiliki nilai *fitness* terkecil, selalu mengganti individu yang paling tua, dan Membandingkan anak dengan kedua orangtua. Apabila anak memiliki nilai *fitness* yang lebih baik daripada salah satu atau kedua orang tua, maka anak menggantikan orang tua yang memiliki nilai *fitness* terendah.

2.8 Kriteria Penghentian

Terdapat berbagai macam kriteria penghentian yang bisa digunakan antara lain (Suyanto,2007:214-215): a) Memberikan batasan jumlah iterasi. Apabila batas iterasi tersebut dicapai, iterasi dihentikan dan laporkan individu bernilai *fitness* tertinggi sebagai solusi terbaik; b) Memberikan batasan waktu proses Algoritma Genetika. Kriteria ini digunakan pada sistem-sistem waktu nyata (*real time systems*), dimana solusi harus ditemukan paling lama misalkan 3 menit. Dengan demikian, Algoritma Genetika bisa dihentikan ketika proses sudah berlangsung selama hampir 3 menit; dan c) Menghitung kegagalan penggantian anggota populasi yang terjadi secara berurutan sampai jumlah tertentu. Misalkan, setelah 100 iterasi tidak ada penggantian individu dalam populasi karena individu anak yang dihasilkan selalu memiliki nilai fitness yang lebih rendah daripada orantuanya. Dalam kondisi seperti ini, kita bisa menghentikan iterasi.

3. METODOLOGI PENELITIAN

3.1 Bentuk Penelitian

Bentuk penelitian yang penulis terapkan adalah studi literatur dengan metode penelitian eksperimental. Eksperimental merupakan metode dengan serangkaian uji coba dilakukan pada setiap bagian perangkat lunak yang telah selesai dibuat, bila bagian tersebut sudah dapat melakukan fungsinya dengan baik, maka uji coba dilakukan pada bagian berikutnya.

3.2 Metode Pengumpulan Data

Didalam mengumpulkan data-data penulis menggunakan metode: 1) Dokumentasi, yaitu mengambil data yang dibutuhkan sebagai data penelitian 2) Studi Literatur, yaitu mempelajari semua literatur yang berkaitan dengan Algoritma Genetika, 3) observasi, yaitu mengamati langsung permainan sudoku yang sudah ada.

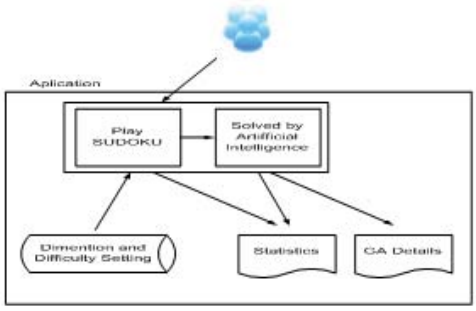
3.3 Metode Pengembangan Perangkat Lunak

Metode yang penulis gunakan dalam merancang perangkat lunak ini adalah menggunakan perancangan *Object Oriented Design.* Bahasa pemrograman yang digunakan adalah Visual Basic 6.

4. HASIL PENELITIAN

4.1 Perancangan Sistem

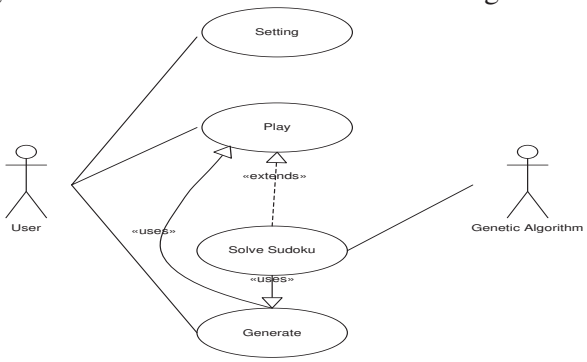
Aplikasi yang dibangun dalam penelitian ini adalah aplikasi Game Sudoku dengan bantuan Algoritma Genetika untuk *generate* dan *solving* Sudoku. ArsitekturAplikasi Game Sudoku dapat dilihat pada gambar 10 dimana saat memainkan game Sudoku, permainan akan langsung menggunakan Algoritma Genetika. Algoritma Genetika digunakan untuk Generate dan Solving Soal, Proses dari algoritma genetika ini akan ditampilkan pada komponen Statistics dan GA Details. Proses secara permainan secara keseluruhan juga akan ditampilkan pada komponen Statistics. Dapat dilihat juga bahwa Setting Dimensi dan tingkat kesulitan akan diambil dari storage setting permainan sebelumnya.



Gambar 10. Arsitektur Aplikasi Game Sudoku

4.2 Use Case Diagram

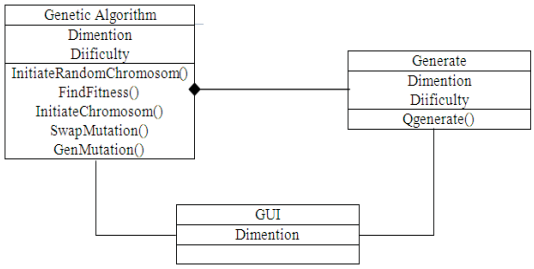
Diagram Use Case yang diperhatikan aktor-aktor yang berperan dalam sistem diperlihat kan ada gambar untuk Actor User dan Genetic Algorithm.



Gambar 11. Use Case dari Actor User dan Genetic Algorithm

4.3 Class Diagram

Class Generate dan *Genetic Algorithm* terdapat hubungan *1 to many* dimana Generate selalu memerlukan Genetic Algorithm untuk menghasilkan soal. Masing-masing *Class* juga terhubung dengan GUI agar dapat difungsikan.

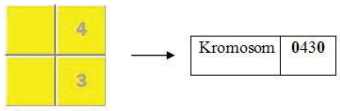


Gambar 12. Rancangan Class Diagram

4.4 Rancangan Algoritma Genetika

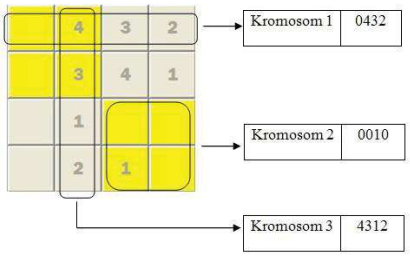
Permainan Sudoku juga menggunakan Algoritma Genetika sebagai Generator Soal dan Solver Soal sudoku. Algoritma Genetika yang digunakan memiliki beberapa tahap, antara lain :

1. terlebih dahulu nilai-nilai dalam cell permainan sudoku ini akan dikonversikan ke dalam gen-gen yang akan membentuk kromosom. Desimal Encoding dipilih karena menyerupai nilai-nilai dalam cell, cell kosong akan dikonversikan menjadi gen 0 dan cell yang memiliki nilai akan langsung diambil nilainya sebagai gen. misalnya :



Gambar 13. Contoh *Desimal Encoding* Game Sudoku

2. Kromosom yang dihasilkan akan merepresentasikan setiap baris, kolom dan region-nya agar optimasi gen pada kromosom-kromosom lebih terarah. Misalnya :



Gambar 14. Contoh representasi kromosom tiap baris, kolom dan region

3. Setelah kromosom dihasilkan, maka dilakukan Inisialisasi nilai awal gen pada kromosom yang bernilai 0 dengan mengacak gen tersebut. Misalnya suatu kromosom memiliki gen **0**32**0**, setelah insialisasi dihasilkan **1**32**4**.
4. Kemudian nilai fitness akan dihitung untuk setiap kromosomnya dengan rumus : $f(x) = \frac{1}{\sum_{i=1}^{n} x_i}$
5. Kromosom akan diurutkan berdasarkan nilai fitness-nya dimulai dari fitness yang bernilai paling kecil karena semakin kecil fitness maka gen-gen dalam kromosom tersebut semakin buruk.
6. Kemudian kromosom akan dimutasi berdasarkan urutan tersebut. Mutasi yang digunakan ada 2 yaitu Mutasi Gen dan Mutasi Swap. Mutasi Gen akan terus

dilakukan selama tidak stuck, ketika stuck maka mutasi swap akan dijalankan. Stuck adalah kondisi dimana mutasi gen tidak lagi menghasilkan kromosom yang lebih baik. Mutasi Gen dilakukan dengan mengganti gen yang dianggap buruk yaitu gen yang memiliki gen lain yang sama dalam 1 kromosom, gen tersebut akan diganti dengan kemungkinan gen yang didapatkan secara acak. Sedangkan Mutasi Swap dilakukan pada gen suatu kromosom yang frekuensi mutasinya paling tinggi, gen tersebut akan ditukar dengan gen lain yang dipilih secara acak.

7. Setelah mutasi selesai, maka proses akan berulang ke perhitungan fitness sampai menghasilkan Solusi yaitu setiap gen dalam kromosom tidak memiliki gen yang sama dalam kromosomnya.

4.5 Implementasi Algoritma Genetika

Algoritma Genetika yang diterapkan pada permainan sudoku guna men-solving soal sudoku. Didalam algoritma genetika terdapat beberapa proses yang dapat dibagi menjadi beberapa fungsi.

Implementasi algoritma genetika pada sudoku dapat dilihat dari Pseudocode dan source code berikut:

```
Mulai
//Tentukan kromosom individu secara acak
Panggil InitiateRandomChromosom
    Lakukan
        //Cari nilai fitness tiap kromosom
        Panggil FindFitness
        //Urutkan kromosom berdasarkan fitness
        Panggil InitiateChromosom
        Jika Stuck Maka
  //Lakukan Mutasi Swap Pada Semua Gen Stuck
            Panggil SwapMutation
        Jika tidak
  //Lakukan Mutasi Gen Pada Semua Gen Buruk
            Panggil GenMutation
        Akhir Jika
     Ulangi Sampai Solution Benar
Selesai
```

Gambar 15. Pseudocode Algoritma Genetika Pada Sudoku

4.6 Pengujian

Pengujian *Solving Time* dilakukan dengan melihat waktu dan generasi yang diperlukan dan untuk men-*generate* soal dan *Solving* soal sudoku. *Generate* soal dilakukan dengan melakukan *solving* cell kosong pada sudoku yang kemudian dihapus secara random. Pengujian akan dilakukan pada Algoritma Genetikanya sehingga hasil yang dilihat hanya pada saat sel kosong selesai di *solving*. Pengujian *Solving cell* kosong ini akan dilakukan sebanyak 10 secara berurutan yang menghasilkan Waktu pemrosesan dan banyak generasi pada sudoku dimensi 4 x 4 dan 9 x 9.

Tabel 1
Pengujian Solving Cell Kosong Dimensi 4 x 4 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	0,032	3
2	0,046	4
3	0,047	5
4	0,047	5
5	0,079	7
6	0,14	14
7	0,156	19
8	0,188	21
9	0,203	22
10	0,328	35

Tabel 2
Pengujian Solving Cell Kosong Dimensi 9 x 9 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	1,609	22
2	2,015	28
3	2,266	31
4	3,141	44
5	4,266	59
6	8,391	119
7	12,515	178
8	18,672	261
9	24	345
10	48,063	681

Pengujian solving juga dilakukan pada tingkat kesulitannya masing-masing untuk dimensi 4 x 4 dan 9 x 9 karena setiap tingkat kesulitan memiliki jumlah Sel kosong yang berbeda yang akan menentukan waktu yang diperlukan untuk *solving*. Pengujian akan dilakukan sebanyak 10 kali secara berurutan untuk masing-masing tingkat kesulitan dan dimensinya. Hasil yang didapat dari pengujian *Solving* adalah Waktu pemrosesan dan Generasi dapat dilihat pada tabel 3, tabel 4, tabel 5, tabel 6, tabel 7 dan tabel 8.

Tabel 3
Pengujian Solving Soal Easy Dimensi 4 x 4 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	0,015	2
2	0,016	2
3	0,016	2
4	0,016	2
5	0,016	2
6	0,016	2
7	0,016	2
8	0,016	2
9	0,016	2
10	0,016	2

Tabel 4
Pengujian Solving Soal Easy Dimensi 9 x 9 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	0,031	2
2	0,032	2
3	0,047	3
4	0,078	4
5	0,093	5
6	0,094	5
7	0,094	5
8	0,14	8
9	1,406	85
10	1,453	86

Tabel 5
Pengujian Solving Soal Medium Dimensi 4 x 4 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	0,015	2
2	0,015	2
3	0,016	2
4	0,016	2
5	0,016	3
6	0,016	3
7	0,063	18
8	0,078	18
9	0,171	51
10	0,177	52

Tabel 6
Pengujian Solving Soal Medium Dimensi 9 x 9 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	0,172	6
2	0,187	7
3	0,187	7
4	0,187	7
5	0,266	10
6	0,344	13
7	0,36	13
8	2,218	84
9	6,203	247
10	11,094	412

Tabel 7
Pengujian Solving Soal Hard Dimensi 4 x 4 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	0,016	3
2	0,016	3
3	0,016	4
4	0,031	5
5	0,031	5
6	0,047	2
7	0,078	18
8	0,172	35
9	0,187	36
10	0,188	34

Tabel 8
Pengujian Solving Soal Hard Dimensi 9 x 9 dengan Algoritma Genetika

No	Waktu (Detik)	Generasi
1	0,344	9
2	0,563	15
3	3,016	85
4	6,594	176
5	9,125	246
6	11,828	330
7	17,844	489
8	25,938	741
9	26,891	735
10	117,672	3005

5 KESIMPULAN

Dari hasil analisis, rancangan penelitian, dan pembahasan dari sejumlah bahasan, maka dapat diambil kesimpulan mengenai Sudoku yang menggunakan Algoritma Genetika yaitu sebagai berikut:

1. Algoritma Genetika dapat digunakan untuk solving permainan sudoku.
2. Aplikasi Sudoku ini menghasilkan Soal sudoku tanpa ada batas.
3. Algoritma Genetika ini menggunakan swap mutation, gen mutation, dan elitism untuk menghasilkan solusi terbaik dengan jumlah generasi yang lebih sedikit.
4. Algoritma Genetika memerlukan waktu yang relatif singkat untuk generate dan Solving.

DAFTAR RUJUKAN

Bennett, S., Mcrabb, S., dan Farmer, R., 2006, *Object Oriented System Analysis and Design using UML*, Mc Graw Hill, New York.

Lopez, E. G., Togelius, J. dan Lucas, S., 2007, *Towards Understanding the Effects of Neutrality on the Sudoku Problem*, England Patent no: 978-1-59593-697-4/07/0007.

Mitchell, Melanie, 1999, *An Intuduction to Genetic Algorithms*, The MIT Press, London.

Nedjah, Nadia, 2006, *Genetic System Programming*, Springer-Verlag Berlin Heidelberg, Netherlands.

Osborne, M. J., dan Rubinstein A., 1994, *A Course In Game Theory*, MIT Press, London.