

Analisis Operator Crossover pada Permasalahan Permainan Puzzle

Kun Siwi Trilestari^[1], *Ade Andri Hendriadi*^[2]

*Program Studi Teknik Informatika, Fakultas Ilmu Komputer,
Universitas Singaperbanga Karawang*

E-mail: siwi_san@yahoo.co.id, hendri2k@gmail.com

ABSTRAK

Puzzle adalah sebuah segiempat dimana ditempatkan beberapa ubin persegiempat. Masing-masing ubin mempunyai nomor di atasnya. Sebuah ubin yang berdekatan dengan ruang kosong dapat digeser ke dalam ruang tersebut. Permainan terdiri dari sebuah posisi awal dan spesifikasi posisi akhir atau tujuan. Dalam menyelesaikan permainan puzzle adalah dengan trial and error. Dengan cara trial and error ini kurang efisien karena waktu yang digunakan dalam menyelesaikan permainan puzzle tersebut cukup lama dan langkah-langkah yang digunakan tidak sistematis yang memungkinkan akan terjadi pengulangan pada langkah yang sama. Salah satu metode yang akan dipakai untuk menyelesaikan permasalahan puzzle ini adalah dengan menggunakan algoritma genetik. Algoritma genetik adalah algoritma pencarian yang bekerja berdasarkan mekanisme seleksi alam dan genetika alam untuk menentukan struktur-struktur atau individu-individu berkualitas tinggi yang terdapat dalam sebuah domain yang disebut populasi. Selain itu, operator crossover merupakan operator terpenting dalam algoritma genetik, karena kombinasi yang dilakukan oleh operator ini merupakan salah satu kekuatan algoritma genetik untuk menyelesaikan masalah. Untuk dapat menyusun puzzle ada beberapa hal yang harus diperhatikan yaitu : representasi kromosom, parameter algoritma genetik seperti P_c , P_m , $P_{popsize}$, dan $Maxgen$.

Kata Kunci: *Puzzle, Algoritma genetik, Crossover*

PENDAHULUAN

Puzzle adalah sebuah segiempat dimana ditempatkan beberapa ubin persegiempat. Masing-masing ubin mempunyai nomor di atasnya. Sebuah ubin yang berdekatan dengan ruang kosong dapat digeser ke dalam ruang tersebut. Permainan terdiri dari sebuah posisi awal dan spesifikasi posisi akhir atau tujuan.

Dalam menyelesaikan permainan puzzle adalah dengan *trial and error*., Dengan cara *trial and error* ini kurang efisien karena waktu yang digunakan dalam menyelesaikan permainan puzzle tersebut cukup lama dan langkah-langkah yang digunakan tidak sistematis yang memungkinkan akan terjadi pengulangan pada langkah yang sama.

Untuk menyelesaikan permasalahan tersebut, pencarian (searching) merupakan salah satu metoda dari pemecahan masalah umum yang sering dipakai.

Adapun salah satu metode yang akan dipakai untuk menyelesaikan permasalahan puzzle ini adalah dengan menggunakan algoritma genetik. Algoritma genetik adalah algoritma pencarian yang bekerja berdasarkan mekanisme seleksi alam dan genetika alam untuk menentukan struktur-struktur atau individu-individu berkualitas tinggi yang terdapat dalam sebuah domain yang disebut populasi. Algoritma genetik tidak sulit dipahami karena algoritma genetik didukung dengan penggunaan perhitungan yang relatif sederhana namun memiliki kekuatan pencarian yang cukup kokoh.

LANDASAN TEORI

Teori yang digunakan dalam masalah ini ini meliputi pencarian Heuristik dimana metode yang digunakan adalah algoritma genetik.

Definisi Puzzle

Puzzle adalah sebuah segiempat dimana ditempatkan beberapa ubin persegiempat. Masing-masing ubin mempunyai nomor di atasnya. Sebuah ubin yang berdekatan dengan ruang kosong dapat digeser ke dalam ruang tersebut. Permainan terdiri dari sebuah posisi awal dan spesifikasi posisi akhir atau tujuan. Tujuan disini adalah untuk mengubah posisi awal menjadi posisi akhir dengan menggeser ubin-ubin di sekelilingnya.

Konfigurasi kotak tersebut (kotak 8, kotak 15, kotak 24, kotak 35, dan kotak 48) dirancang menggunakan komponen-komponen visual yang setiap angkanya dapat digeser-geser atau diubah-ubah seperti permainan kotak sesungguhnya.

Konfigurasi tersebut disimpan dalam struktur larik (array) 2 dimensi atau matriks untuk mendefinisikan susunan angka 1, 2, 3, 4, 5, 6, 7, 8 dan area kosong. Angka-angka yang dapat digeser adalah angka-angka yang berdekatan dengan area kosong.

Definisi Algoritma Genetik

Algoritma genetik adalah algoritma pencarian heuristik yang didasarkan atas mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dan kromosom antar individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidup. Pada dasarnya ada empat kondisi yang sangat mempengaruhi proses evaluasi yaitu :

1. Kemampuan organisme untuk melakukan reproduksi
2. Keberadaan populasi organisme yang bisa melakukan reproduksi
3. Keberagaman organisme dalam suatu populasi
4. Perbedaan kemampuan untuk survive

Individu yang lebih kuat (fit) akan memiliki tingkat survival dan tingkat reproduksi yang lebih tinggi jika dibandingkan dengan individu yang kurang fit. Pada kurun waktu tertentu (sering dikenal dengan istilah generasi), populasi secara keseluruhan akan lebih banyak memuat organisme yang fit.

Algoritma Genetik pertama kali dikembangkan oleh John Holland dari universitas Michigan (1975). John Holland mengatakan bahwa setiap masalah yang berbentuk adaptasi (alami maupun buatan) dapat diformulasikan dalam terminologi genetika. Algoritma genetik adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom.

Struktur Umum Algoritma Genetik

Pada algoritma genetik ini, teknik pencarian dilakukan sekaligus atas sejumlah solusi yang mungkin yang dikenal dengan populasi. Individu yang terdapat dalam satu populasi disebut istilah *kromosom*. Kromosom ini merupakan suatu solusi yang masih berbentuk simbol. Populai awal dibangun secara acak, sedangkan populasi berikutnya merupakan hasil evolusi kromosom-kromosom melalui iterasi yang disebut dengan istilah *generasi*. Pada setiap generasi, kromosom akan melalui proses evaluasi dengan menggunakan alat ukur yang disebut dengan fungsi *fitness*. Nilai fitness dari suatu kromosom akan menunjukkan kualitas kromosom dalam populasi tersebut. Generasi berikut dikenal dengan istilah anak (*offspring*) terbentuk dari gabungan dua kromosom generasi sekarang yang bertindak sebagai induk (*parent*) dengan menggunakan operator penyilangan (*crossover*). Selain operator penyilangan, suatu kromosom juga dapat dimodifikasi dengan menggunakan operator *mutasi*. Populasi generasi yang baru dibentuk dengan cara menyeleksi nilai fitness dari kromosom induk (*parent*) dan nilai fitness dari kromosom anak (*offspring*), serta menolak kromosom-kromosom yang lainnya sehingga ukuran populasi (jumlah kromosom dalam suatu populasi) konstan. Setelah melalui beberapa generasi, maka algoritma ini akan konvergen ke kromosom terbaik.

Algoritma Genetik Sederhana

Misalkan $P(\text{generasi})$ adalah populasi dari suatu generasi, maka secara sederhana algoritma genetik terdiri dari langkah-langkah :

1. Generasi = 0 (generasi awal)
2. Inisialisasi populasi awal, $P(\text{generasi})$, secara acak
3. Evaluasi nilai fitness pada setiap individu dalam $P(\text{generasi})$
4. Kerjakan langkah-langkah berikut sehingga generasi mencapai maksimum generasi :
 - a. Generasi = generasi + 1 (tambah generasi)
 - b. Seleksi populasi tersebut untuk mendapatkan kandidat induk, $P'(\text{generasi})$
 - c. Lakukan crossover pada $P'(\text{generasi})$
 - d. Lakukan mutasi pada $P'(\text{generasi})$
 - e. Lakukan evaluasi fitness setiap individu pada $P'(\text{generasi})$
 - f. Bentuk populasi baru : $P(\text{generasi}) = \{P(\text{generasi} - 1) \text{ yang survive, } P'(\text{generasi})\}$

Operator Genetik Crossover

Crossover adalah suatu implementasi buatan yang menyangkut pertukaran dari informasi genetic yang terjadi didalam real-life reproduksi. Dalam crossover melibatkan dua kromosom sebagai parent yang nantinya akan disilangkan untuk mendapatkan *offspring* baru. Jenis crossover pada algoritma genetic adalah :

Single Point Crossover, Two Point Crossover, Uniform Crossover, dan Permutasi Crossover.

Rancangan Sistem-Identifikasi Masalah

Masalah yang akan dipaparkan adalah permainan puzzle. Puzzle adalah sebuah segiempat dimana ditempatkan beberapa ubin persegiempat. Masing-masing ubin mempunyai nomor di atasnya. Sebuah ubin yang berdekatan dengan ruang kosong dapat digeser ke dalam ruang tersebut. Permainan terdiri dari sebuah posisi awal dan spesifikasi posisi akhir atau tujuan. Tujuan disini adalah untuk mengubah posisi awal menjadi posisi akhir dengan menggeser ubin-ubin di sekelilingnya. Konfigurasi kotak 8, 15, 24, 35, dan 48 dirancang menggunakan komponen-komponen visual yang setiap angkanya dapat digeser-geser atau diubah-ubah seperti permainan kotak sesungguhnya.

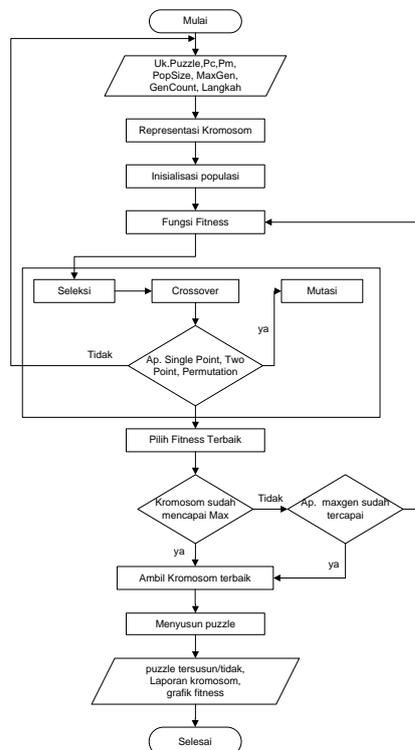
Konfigurasi kotak disimpan dalam suatu matriks sebagai berikut :

Untuk Puzzle 8 yaitu :

Matrix = array[0..2,0..2] of byte untuk mendefinisikan setiap susunan angka dan area kosong.

Dalam data matriks tersebut akan merepresentasikan susunan angka-angka dalam kotak 8, 15, 24, 35 dan 48. Kotak kosong untuk merepresentasikan area kosong. Angka-angka yang dapat digeser adalah angka-angka yang berdekatan dengan area kosong (vertikal-horizontal).

Penyelesaian Masalah dengan Algoritma Genetik



Gambar 1. Diagram alir penyelesaian algoritma genetik

Implementasi

Pada bab ini akan membahas mengenai implementasi dari algoritma genetik dalam menyelesaikan permainan puzzle, dengan menggunakan operator genetik crossover : *Single Point Crossover*, *Two Point Crossover*, *Permutation*. Seleksi yang digunakan ada dua, yaitu *Rank Selection* dan *Roulette Wheel*. Setelah semua rancangan selesai dibuat dan hal-hal yang diperlukan dalam pembuatan program sudah diperoleh, maka dibuatlah sebuah program aplikasi algoritma genetik untuk memvisualisasikan permainan puzzle dengan menggunakan bahasa pemrograman Delphi 7.0, bekerja sesuai dengan adanya parameter.

Hasil Uji Coba

Berikut adalah hasil percobaan yang dilakukan terhadap permasalahan permainan puzzle dengan ukuran puzzle yang berbeda-beda.

Pengujian Tahap Pertama untuk Menentukan Parameter Terbaik

Percobaan dilakukan dengan ukuran populasi 500, maxgen 50, Probabilitas Crossover (Pc) 0,9, Probabilitas Mutasi (Pm) 0,03, Gen Count sebanyak 25, dan langkah pengacakan sebanyak 10.

Tabel 1. hasil uji coba

| Generasi | Ukuran Puzzle | Generasi Terbaik | Fitness | Jumlah Langkah |
|----------|---------------|------------------|---------|----------------|
| 50 | 3x3 | 6 | 8 | 12 |
| 50 | 4x4 | 3 | 15 | 25 |
| 50 | 5x5 | 6 | 24 | 24 |
| 50 | 6x6 | 6 | 35 | 17 |
| 50 | 7x7 | 11 | 48 | 24 |

Dari hasil percobaan tahap pertama tersebut, pengujian sistem dengan penentuan parameter algoritma genetik Pc = 0,9; Pm = 0,03; Popsize = 500; Maxgen = 50, dan *Gen Count* (jumlah gen / jumlah langkah) = 25 dapat memberikan solusi yang optimum dalam menyusun permainan puzzle.

Pengujian Tahap Kedua untuk Menentukan Waktu Keberhasilan

Berikut adalah hasil percobaan tahap kedua yang dilakukan terhadap permasalahan puzzle dengan jenis operator yang berbeda, penggunaan seleksi yang berbeda dan ukuran puzzle 3 x 3. Percobaan dilakukan dengan ukuran populasi 500, maksimum generasi 50, Gen Count = 100 langkah pengacakan sebanyak 10, sedangkan Pc = 0,9 dan Pm = 0,03.

Tabel 2. Pengujian dengan menggunakan Rank Selection dan Single Point Crossover

pada puzzle berukuran 3 x 3

| Pengujian | Generasi Terbaik | Fitness | Waktu (/dtk) |
|-----------|------------------|---------|--------------|
| 1 | 9 | 8 | 1 |
| 2 | 1 | 8 | 0 |
| 3 | 4 | 8 | 1 |
| 4 | 1 | 8 | 0 |

| | | | |
|----|---|---|---|
| 5 | 2 | 8 | 1 |
| 6 | 3 | 8 | 1 |
| 7 | 1 | 8 | 0 |
| 8 | 5 | 8 | 1 |
| 9 | 1 | 8 | 0 |
| 10 | 4 | 8 | 1 |

Berdasarkan hasil dari 20 pengujian untuk ukuran puzzle 3x3, terdapat 20 (100 %) pengujian yang berhasil dan 0 (0 %) pengujian yang gagal. Dengan catatan waktu ± 1 detik.

Tabel 3. Pengujian dengan menggunakan Rank Selection dan Two Point Crossover pada puzzle berukuran 3 x 3

| Pengujian | Generasi Terbaik | Fitness | Waktu (/dtk) |
|-----------|------------------|---------|--------------|
| 1 | 5 | 8 | 1 |
| 2 | 1 | 8 | 1 |
| 3 | 4 | 8 | 0 |
| 4 | 1 | 8 | 0 |
| 5 | 1 | 8 | 0 |
| 6 | 1 | 8 | 0 |
| 7 | 6 | 8 | 1 |
| 8 | 1 | 8 | 0 |
| 9 | 2 | 8 | 0 |
| 10 | 1 | 8 | 0 |

Berdasarkan hasil dari 20 pengujian untuk ukuran puzzle 3x3, terdapat 20 (100 %) pengujian yang berhasil dan 0 (0 %) pengujian yang gagal. Dengan catatan waktu ± 1 detik.

Tabel 4. Pengujian dengan menggunakan Rank Selection dan Permutasi Crossover pada puzzle berukuran 3 x 3

| Pengujian | Generasi Terbaik | Fitness | Waktu (/dtk) |
|-----------|------------------|---------|--------------|
| 1 | 8 | 8 | 1 |
| 2 | 1 | 8 | 1 |
| 3 | 1 | 8 | 0 |
| 4 | 1 | 8 | 0 |
| 5 | 4 | 8 | 0 |
| 6 | 5 | 8 | 1 |
| 7 | 1 | 8 | 0 |
| 8 | 3 | 8 | 0 |

| | | | |
|----|---|---|---|
| 9 | 3 | 8 | 1 |
| 10 | 1 | 8 | 1 |

Berdasarkan hasil dari 20 pengujian untuk ukuran puzzle 3x3, terdapat 20 (100 %) pengujian yang berhasil dan 0 (0 %) pengujian yang gagal. Dengan catatan waktu ± 1 detik.

Tabel 5. Pengujian dengan menggunakan Roulette Wheel Selection dan Single Point Crossover pada puzzle berukuran 3 x 3

| Pengujian | Generasi Terbaik | Fitness | Waktu (/dtk) |
|-----------|------------------|---------|--------------|
| 1 | 2 | 8 | 0 |
| 2 | 2 | 8 | 1 |
| 3 | 1 | 8 | 0 |
| 4 | 1 | 8 | 1 |
| 5 | 2 | 8 | 0 |
| 6 | 4 | 8 | 0 |
| 7 | 1 | 8 | 1 |
| 8 | 2 | 8 | 0 |
| 9 | 1 | 8 | 0 |
| 10 | 1 | 8 | 0 |

Berdasarkan hasil dari 20 pengujian untuk ukuran puzzle 3x3, terdapat 20 (100 %) pengujian yang berhasil dan 0 (0 %) pengujian yang gagal. Dengan catatan waktu ± 1 detik.

Tabel 6. Pengujian dengan menggunakan Roulette Wheel Selection dan Two Point Crossover pada puzzle berukuran 3 x 3

| Pengujian | Generasi Terbaik | Fitness | Waktu (/dtk) |
|-----------|------------------|---------|--------------|
| 1 | 2 | 8 | 0 |
| 2 | 1 | 8 | 0 |
| 3 | 2 | 8 | 1 |
| 4 | 1 | 8 | 0 |
| 5 | 2 | 8 | 0 |
| 6 | 2 | 8 | 0 |
| 7 | 3 | 8 | 0 |
| 8 | 1 | 8 | 1 |
| 9 | 1 | 8 | 0 |
| 10 | 1 | 8 | 1 |

Berdasarkan hasil dari 20 pengujian untuk ukuran puzzle 3x3, terdapat 20 (100 %) pengujian yang berhasil dan 0 (0 %) pengujian yang gagal. Dengan catatan waktu ± 1 detik.

Tabel 7. Pengujian dengan menggunakan Roulette Wheel Selection dan Permutasi Crossover pada puzzle berukuran 3 x 3

| Pengujian | Generasi Terbaik | Fitness | Waktu (/dtk) |
|-----------|------------------|---------|--------------|
| 1 | 4 | 8 | 0 |
| 2 | 1 | 8 | 0 |
| 3 | 3 | 8 | 1 |
| 4 | 5 | 8 | 0 |
| 5 | 1 | 8 | 0 |
| 6 | 4 | 8 | 0 |
| 7 | 2 | 8 | 0 |
| 8 | 1 | 8 | 1 |
| 9 | 2 | 8 | 0 |
| 10 | 6 | 8 | 1 |

Berdasarkan hasil dari 20 pengujian untuk ukuran puzzle 3x3, terdapat 20 (100 %) pengujian yang berhasil dan 0 (0 %) pengujian yang gagal. Dengan catatan waktu ± 1 detik.

KESIMPULAN

Berdasarkan tujuan dalam penulisan skripsi ini, maka dapat disimpulkan bahwa :

1. Berdasarkan hasil percobaan pengujian sistem dengan penentuan parameter algoritma genetik $P_c = 0,9$; $P_m = 0,03$; $P_{opsize} = 500$; $Maxgen = 50$, $Gen\ Count = 25$ dan langkah pengacakan sebanyak 10 dalam satu kali percobaan, diperoleh Seleksi yang terbaik adalah *Roulette Wheel* dan Operator genetik crossover yang terbaik adalah *Single Point Crossover* dalam menyusun permainan puzzle pada masing-masing ukuran. Semua itu berdasarkan catatan waktu yang diperlukan untuk mengeksekusi program puzzle.
2. Berdasarkan hasil pengujian pada tahap kedua, algoritma genetik mampu menyelesaikan permainan puzzle pada ukuran puzzle 3x3, 4x4, 5x5, 6x6, dan 7x7 dengan hasil yang sangat optimum dengan tingkat keberhasilan sebesar 100 %. Sehingga dapat diketahui bahwa seleksi yang terbaik adalah *Roulette Wheel* dan Operator genetik crossover yang terbaik adalah *Single Point Crossover*.
3. Semakin besar Ukuran Puzzle, maka waktu yang dibutuhkan untuk mengeksekusi program akan semakin lama.
4. Berdasarkan hasil percobaan tahap kedua, algoritma genetik baik digunakan pada permasalahan dengan ruang solusi yang lebih besar.

SARAN

Berdasarkan pada proses yang dilakukan masih banyak kekurangan dan kelemahan sehingga perlu dikembangkan lagi agar kinerjanya lebih baik, oleh karena itu penulis memeberikan beberapa masukan berupa saran-saran sebagai berikut :

1. Permainan puzzle ini dapat ditambah ukuran puzzlenya menjadi lebih besar dan bentuk kotaknya dapat diubah menjadi segitiga ataupun yang lainnya sesuai dengan keinginan.
2. Implementasi kedalam bahasa pemrograman yang berbeda.
3. Implementasi pada permainan yang lain.
4. Pembuatan representasi kromosom untuk pemecahan permasalahan permainan puzzle agar menghasilkan solusi yang lebih baik.

DAFTAR PUSTAKA

- Divisi Penelitian dan Pengembangan MADCOMS. (2002). *Seri Panduan Pemrograman Borland Delphi 7*. Jilid 1. Yogyakarta: Andi Offset.
- Gen, Mitsuo and Chang, Runwei. (1997). *Genetic Algorithms & Engineering Design*. New York: Jhon Wiley and Soons.
- Goldberg, David E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. New York: Addison Wesley Publishing.
- Kusumasewi, Sri, & Purnomo, Hari. (2005). *Penyelesaian Masalah Optimasi dengan Teknik-teknik Heuristik*. Edisi Pertama. Yogyakarta: Penerbit Graha Ilmu.
- Michalewicz, Zbigniew. (1992). *Genetic Algorithms + Data Structures = Evolution Program*. Second Extended Edition. Springer – Verlag.