PENYELESAIAN *JIGSAW PUZZLE* DENGAN ALGORITMA RUNUT-BALIK

Rianto Fendy Kristanto - 13507036

Jurusan Teknik Informatika Institut Teknologi Bandung
Jalan Ganesha 10 Bandung, Jawa Barat 40132
e-mail: if17036@students.if.itb.ac.id

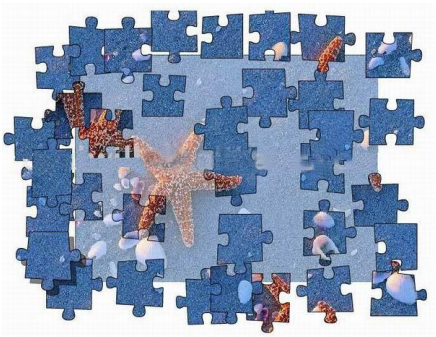
ABSTRAK

***Jigsaw puzzle* merupakan salah satu permainan menyusun potongan-potongan *piece* pada posisi yang benar sehingga potongan-potongan *piece* tersebut menjadi sebuah gambar. *Jigsaw puzzle* memiliki jumlah *piece* yang nantinya harus disusun untuk menjadi gambar dan dimensi, yaitu panjang dan lebar. Saat ini, terdapat jigsaw puzzle 2 dimensi (2D) dan 3 dimensi (3D). Jumlah *piece* seringkali disebut sebagai ukuran *jigsaw puzzle*. Setiap *piece* memiliki bentuk yang menjadi kunci untuk dipasangkan dengan beberapa *piece* lainnya yang memiliki bentuk yang cocok untuk dipasangkan dengan bentuk *piece* tersebut. Jadi setiap *piece* memiliki pasangan *piece-piece* yang tetap untuk disusun. Penyelesaikan *jigsaw puzzle* dapat menggunakan metode di dalam strategi algoritma. Algoritma-algoritma yang mungkin dapat dipakai untuk menyelesaikan *jigsaw puzzle* adalah algoritma *brute force*, algoritma BFS (*Breadth First Search*), algoritma DFS (*Depth First Search*), algoritma runut-balik (*backtracking*), dan masih banyak lagi. Dalam makalah ini, penulis akan menggunakan algoritma runut-balik untuk menyelesaikan *jigsaw puzzle*. Algoritma runut-balik adalah algoritma yang berbasis pada DFS untuk mencari solusi persoalan secara lebih mangkus. Algoritma runut-balik hanya mencari solusi persoalan di antara semua kemungkinan solusi yang ada. Hanya pencaian yang mengarah ke solusi saja yang perlu dipertimbangkan. Akibatnya, waktu pencarian dapat dihemat [4].**

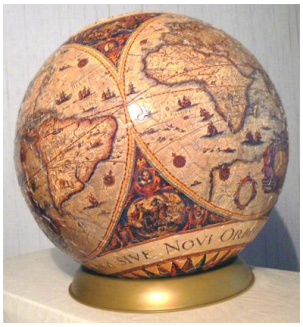
Kata kunci: *jigsaw puzzle*, runut-balik, *piece*, solusi.

1. PENDAHULUAN

Jigsaw Puzzle pertama kali diproduksi oleh John Spilsbury pada tahun 1760-an [5]. *Jigsaw puzzle* merupakan sebuah permainan menyusun potongan-potongan *piece* pada posisi yang benar sehingga potongan-potongan *piece* tersebut menjadi sebuah gambar. *Jigsaw puzzle* memiliki ukuran, yaitu jumlah *piece*-nya, dan dimensi, yaitu panjang dan lebar. Ukuran-ukuran yang sering muncul adalah 300-*piece*, 500-*piece*, 1000-*piece*, hingga lebih dari 5000-*piece*. Tetapi saat ini, *jigsaw puzzle* ada yang 2 dimensi (2D) dan ada yang 3 dimensi (3D). *Jigsaw puzzle* yang 3 dimensi disebut juga *puzzle globe* [2]. *Jigsaw puzzle* 2 dimensi biasanya lebih sering dimainkan daripada *jigsaw puzzle* 3 dimensi.



Gambar 1. Jigsaw Puzzle 2dimensi [1]



Gambar 2. Jigsaw Puzzle 3 dimensi [3]

Setiap *piece* pada *jigsaw puzzle* memiliki bentuk yang menjadi kunci untuk dipasangkan dengan beberapa *piece* lainnya yang memiliki bentuk yang cocok untuk dipasangkan dengan *piece* tersebut. Sehingga, kita menggunakan strategi-strategi khusus untuk menyelesaikan *jigsaw puzzle*. Strategi yang biasa digunakan adalah menyusun *piece-piece* mulai dari pinggiran hingga ke bagian dalam *jigsaw puzzle*. Strategi lainnya adalah melihat gambar yang terdapat pada setiap *piece* dan mengelompokan *piece-piece* yang memiliki gambar hampir serupa kemudian menyusun *piece-piece* yang memiliki gambar serupa tersebut hingga menjadi

bagian kecil dari gambar *jigsaw puzzle*. Bagian kecil tersebut dianggap sebagai *piece* baru yang memiliki ukuran lebih besar. Ulangi pengelompokan *piece-piece* tersebut dan penyusunan *piece-piece* dengan gambar yang serupa hingga menjadi satu *piece* yang besar yaitu gambar *jigsaw puzzle*.

Pendekatan metode algoritma yang digunakan untuk menyelesaikan *jigsaw puzzle* akan berbeda-beda untuk setiap algoritma yang dipakai. Dalam makalah ini, penulis menggunakan algoritma runut-balik (*backtracking*) untuk menyelesaikan *jigsaw puzzle*. Tujuan akhir dari penyelesaian *jigsaw puzzle* adalah mendapatkan gambar *jigsaw puzzle*, sedangkan kondisi awal adalah semua *piece* pada *jigsaw puzzle* tidak terpasangkan dengan *piece* lainnya. Tujuan akhir merupakan solusi yang ingin dicapai. Kondisi awal merupakan status awal dari penyelesaian *jigsaw puzzle* dengan algoritma runut-balik.

2. ALGORITMA RUNUT-BALIK

Algoritma runut-balik (*backtracking*) adalah algoritma yang berbasis pada DFS untuk mencari solusi persoalan secara lebih mangkus. Algoritma runut-balik secara sistematis hanya mencari solusi persoalan di antara semua kemungkinan solusi yang ada. Hanya pencarian yang mengarah ke solusi saja yang perlu dipertimbangkan. Akibatnya, waktu pencarian dapat dihemat [4].

2.1 Properti Umum Algoritma Runut-Balik

Untuk menerapkan algoritma runut-balik, properti-properti berikut perlu didefinisikan:

1. Solusi persoalan

Solusi persoalan dinyatakan sebagai vektor dengan n-*tuple*:

$$X = (x_1,x_2,x_3, ..., x_n) , x_i \in \text{himpunan berhingga Si} \quad (1)$$

Contoh: $S_i = \{ 0 , 1 \}$

$x_i = 0$ atau 1

2. Fungsi pembangkit nilai x_k

Fungsi pembangkit nilai x_k dinyatakan sebagai: T(k).

T(k) membangkitkan niali untuk x_k, yang merupakan komponen vektor solusi.

3. Fungsi pembatas

Fungsi pembatas dinyatakan sebagai:

$B(x_1,x_2,x_3, ..., x_k)$

$B(x_1,x_2,x_3, ..., x_k)$ menyatakan apakah $(x_1,x_2,x_3, ..., x_k)$ mengarah ke solusi. Jika ya, maka pembangkitan nilai untuk x_{k+1} dilanjutkan, tetapi jika tidak, maka $(x_1,x_2,x_3, ..., x_k)$ dibuang dan tidak dipertimbangkan lagi dalam pencarian solusi.

Selain penentuan ketiga properti, yaitu solusi persoalan, fungsi pembangkit, dan fungsi pembatas, kita perlu mendefinisikan semua kemungkinan solusi dari persoalan disebut ruang solusi [4]. Ruang solusi dapat dinyatakan dengan:

$$S_1 \times S_2 \times S_3 \times S_4 \times ... \times S_n, \text{ untuk } x_i \in S_i \quad (2)$$

Jumlah ruang solusi adalah:

$$|S_1| . |S_2| . |S_3| . |S_4| . ... . |S_n| \quad (3)$$

2.2 Prinsip Pencarian Solusi dengan Algoritma Runut-Balik

Langkah-langkah pencarian solusi dengan algoritma runut-balik adalah sebagai berikut:

1. Solusi dicari dengan membentuk lintasan dari akar ke daun. Aturan pembentukkan simpul yang dipakai adalah mengikuti metode pencarian mendalam (DFS). Simpul-simpul yang sudah dilahirkan dinamakan dengan simpul hidup. Simpul hidup yang sedang diperluas dinamakan simpul *expand*. Simpul dinomori dari atas ke bawah sesuai dengan urutan pembentukkan simpul.
2. Setiap simpul *expand* diperluas, lintasan yang dibangun olehnya semakin bertambah panjang. Jika lintasan yang sedang dibentuk tidak mengarah ke solusi, maka simpul *expand* "dibunuh" sehingga menjadi simpul mati. Fungsi yang digunakan untuk membunuh simpul *expand* adalah fungsi pembatas. Perlu diingat, simpul mati tidak perlu diperluas lagi.
3. Jika pembentukkan lintasan berakhir dengan simpul mati, maka pencarian solusi diteruskan dengan pembangkitan simpul anak yang lain. Jika tidak ada lagi simpul anak yang dapat dibangkitkan, maka pencarian solusi dilanjutkan dengan melakukan runut-balik ke simpul hidup lainnya, yaitu simpul orang tua (simpul orang tua pasti merupakan simpul hidup). Selanjutnya simpul orang tua menjadi simpul *expand* dan memperluas lagi simpul *expand* tersebut.
4. Pencarian dihentikan bila solusi telah ditemukan atau tidak ada lagi simpul yang dapat dibangkitkan.

2.3 Penggunaan Algoritma Runut-Balik dalam Menyelesaikan *Jigsaw Puzzle*

Asumsi-asumsi yang dipakai untuk menyelesaikan *jigsaw puzzle* dengan algoritma runut-balik adalah:

1. Kondisi awal atau status awal (simpul akar dari pohon ruang status) adalah tidak ada *piece* pada *jigsaw puzzle* yang terpasangkan dengan *piece* lainnya.

2. Solusi yang ingin dicapai adalah seluruh *piece jigsaw puzzle* terpasangkan dengan *piece-piece* lainnya sehingga terbentuk satu *piece* yang besar dan terbentuk gambar dari *jigsaw puzzle*.

3. Ukuran *jigsaw puzzle* adalah sebanyak n.

4. Semua *piece jigsaw puzzle* dinomori secara acak mulai dari 1 hingga ke *n*. *Piece* yang diberi nomor 1 akan disebut *piece* 1, *piece* yang diberi nomor 2 akan disebut *piece* 2, dan begitu setersunya.

5. Semua posisi *jigsaw puzzle* dinomori secara acak mulai dari 1 hingga ke *n*. Posisi yang diberi nomor 1 akan disebut posisi 1, posisi yang diberi nomor 2 akan disebut posisi 2, dan begitu setersunya.

6. *Piece* 1 dikatakan dapat dipasangkan dengan *piece* 2 jika *piece* 1 memiliki bentuk yang dapat dipasangkan dengan *piece* 2 dan gambar antara *piece* 1 dan *piece* 2 saling berhubungan.

7. Jika *piece* 1 dapat dipasangkan dengan *piece* 2, maka *piece* 2 dapat dipasangkan dengan *piece* 1.

8. Posisi(*k*) merupakan posisi yang ditempati oleh *piece k*.

Misalkan solusi dinyatakan sebagai vektor X dengan *n*-tuple:

$$X = (x_1,x_2,x_3,x_4, ..., x_n),\ x_i \in \{ 1, 2, 3, 4, ..., n \} \quad (4)$$

dengan *n* adalah ukuran *jigsaw puzzle*. $x_k = 1$, artinya posisi k pada *jigsaw puzzle* ditempati oleh piece 1.

Fungsi pembangkit *piece k* adalah:

$$T(k) : x_{posisi(k)} \leftarrow k \quad (5)$$

Fungsi pembatasnya adalah:

$B(x_1,x_2,x_3, ..., x_k)$:
(*piece* k pada posisi x_k dapat dipasangkan dengan *piece* pada x_1 atau *piece* x_2 atau *piece* x_3 atau ... atau *piece* x_{k-1})
(6)

Langkah-langkah penyelesaian *jigsaw puzzle* dengan menggunakan algoritma runut-balik adalah:

1. Ambil sembarang *piece k* dan letakkan pada posisi 1. Isilah x_1 dengan *k*.
2. Kemudian bangkitkan *piece k* lainnya dengan fungsi pembangkit untuk dipasangkan dengan piece-piece yang sudah terpasang dengan benar.
3. Jika *piece k* tidak dapat dipasangkan pada *piece-piece* yang sudah ada, maka bangkitan *piece k* berikutnya dengan fungsi pembangkit, ulangi langkah 2.
4. Jika tidak ada *piece k* yang dapat dibangkitkan, maka penyelesaian *jigsaw puzzle* tidak dapat diselesaikan.
5. Jika *piece k* dapat dipasangkan pada *piece-piece* yang sudah ada, isilah $x_{posisi(k)}$ dengan *k*. Kemudian bangkitkan *piece k* lainnya dengan fungsi pembangkit, ulangi langkah 2.
6. Jika tidak ada *piece k* yang dapat dibangkitkan dan *k* = *n*, maka solusinya *jigsaw puzzle* ditemukan.

3. KESIMPULAN

Kesimpulan yang dapat diambil dari makalah ini adalah:

1. Penyelesaian *jigsaw puzzle* memiliki strategi-strategi tersendiri. Misalnya dengan menyusun *piece-piece* mulai dari pinggiran hingga ke bagian dalam *jigsaw puzzle*.
2. Penyelesaian *jigsaw puzzle* dapat menggunakan algoritma runut-balik.

REFERENSI

[1] Jigsaw Puzzle (502) - China jigsaw puzzle,toy puzzle,puzzles in Intellectual & Educational Toys, http://dgweida.en.made-in-china.com/product/bMaQuKPzXDVd/China-Jigsaw-Puzzle-502-.html, diakses tanggal 5 Januari 2010 pukul 20.00.

[2] Jigsaw Puzzle: Definition from Answers.com, http://www.answers.com/topic/jigsaw-puzzle, diakses tanggal 5 Januari 2010 pukul 18.00.

[3] File:3d-jigsaw-puzzle.jpg - Wikimedia Commons, http://commons.wikimedia.org/wiki/File:3d-jigsaw-puzzle.jpg, diakses tanggal 5 Januari 2010 pukul 20.23.

[4] Munir, Rinaldi, "Diktat Kuliah IF3051 Strategi Algoritma", Program Studi Teknik Informatika, 2009.

[5] Puzzle History, http://www.jigsaw-puzzle.org/jigsaw-puzzle-history.html, diakses tanggal 5 Januari 2010 pukul 18.00.