# Penerapan Algoritma Exact Cover Problem pada Persoalan Pentomino Puzzle

Adistya Alindita[1], R Audi Primadhanty[2], Lely Triastiti[3]

*Departemen Teknik Informatika*
*Institut Teknologi Bandung*
*Jl. Ganesha 10, Bandung*

E-mail : if13116@students.if.itb.ac.id[1], if13102@students.if.itb.ac.id[2], if13070@students.if.itb.ac.id[3]

## Abstrak

Puzzle merupakan suatu bentuk permainan yang cukup dikenal masyarakat. Salah satu bentuk permainan puzzle yang menarik adalah Pentomino Puzzle. Pentomino sendiri merupakan istilah untuk menggambarkan bentuk bangun ruang dua dimensi yang disusun dari lima bujur sangkar berukuran sama. Ada dua belas kemungkinan bentuk pentomino yang dapat tercipta[3]. Inti dari permainan Pentomino Puzzle adalah bagaimana seorang pemain menempatkan kedua belas pentomino sehingga membentuk suatu bangun ruang dua dimensi yang diinginkan. Kreativitas seorang pemain sangat dibutuhkan dalam memecahkan persoalan-persoalan Pentomino Puzzle. Seringkali permainan ini diselipkan dalam tes *aptitude* atau IQ pada anak. Pemecahan masalah permainan Pentomino Puzzle dapat diselesaikan dengan beberpa algoritma. Hingga saat ini solusi sederhana yang paling mangkus untuk persoalan Pentomino Puzzle adalah dengan menerapkan algoritma DFS dengan *backtracking*. Donald E. Knuth menunjukkan bahwa pemecahan dari masalah ini, dengan merujuk persoalan yang lebih umum, adalah dengan menggunakan *Exact Cover Problem*. Dengan demikian, kita dapat menggunakan algoritma umum dari *Exact Cover Problem* dan menggunakannnya untuk memecahkan persoalan Pentomino Puzzle. Dalam makalah ini akan dibahas tentang langlah-langkah penerapan algoritma *Exact Cover Problem* yang merupakan gabungan Depth First Search (DFS) dan *backtracking* untuk menyelesaikan persoalan Pentomino Puzzle.

***Kata kunci**: Pentomino Puzzle, Exact Cover Problem, DFS, backtracking, algoritma*

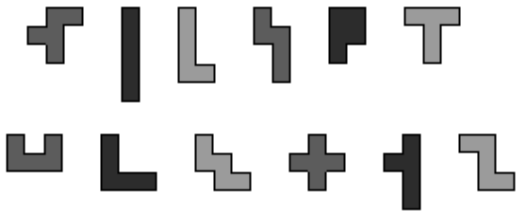
## 1. Pendahuluan

Permainan Pentomino Puzzle adalah permainan yang cukup sering digunakan pada tes *aptitude* atau tes IQ pada anak. Dalam permainan ini, seorang pemain dituntut untuk membentuk suatu bangun ruang dua dimensi hanya dengan menggunakan dua belas potong pentomino yang ada. Kadang kala bangun ruang yang harus disusun cukup rumit bentuknya sehingga persoalannya menjadi sulit untuk dipecahkan. Untuk beberapa persoalan yang rumit mungkin tidak ditemukan solusinya. Melalui permainan ini, seorang pemain dapat diasah kemampuan berpikir dan kreativitasnya.

## 2. Pentomino Puzzle

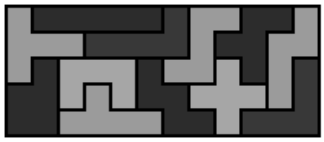
Pentomino Puzzle adalah satu set puzzle yang terdiri dari dua belas potongan pentomino. Tiap pentomino merupakan hasil penyusunan lima bujur sangkar berukuran sama sehingga membentuk bangun ruang dua dimensi yang berbeda-beda [2].

Keduabelas bentuk dari pentomino yang ada adalah sebagai berikut :

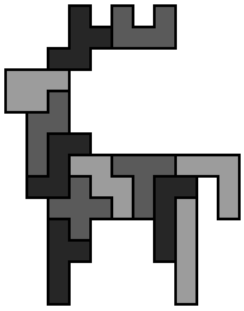


Potongan-potongan pentomino ini kemudian dapat disusun untuk menciptakan beraneka ragam bentuk yang lebih besar.

Contoh permainannya adalah menyusun dua belas pentomino sehingga membentuk persegi panjang berukuran 5x12. Salah satu solusi penyusunannya apat dilihat di gambar berikut :



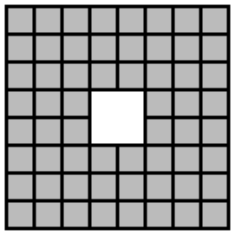
Contoh lain adalah menyusun dua belas pentomino sehingga membentuk rusa seperti berikut :



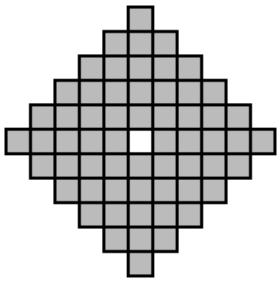
Umumnya persoalan yang rumit dimulai dengan penyusunan persegi panjang berisi 60 bujur sangkar yang kosong, sehingga kita harus mengisinya hingga tepat tertutup semua permukaannya.

## 3. Penyelesaian Masalah Pentomino Puzzle

Salah satu persoalan yang menarik adalah bagaimana seorang pemain menyelesaikan Pentomino Puzzle untuk bujur sangkar berukuran 8x8 dengan lubang 2x2 di tengah, seperti tampak pada gambar berikut :



Persoalan lain yang tak kalah menarik adalah menetukan solusi untuk membentuk bangun diamond dengan 60 bujur sangkar seperti berikut :



Untuk memecahkan persoalan-persoalan tersebut kita dapat menerapkan algoritma Depth First Search (DFS).

### 3.1 Penggunaan Algoritma DFS

Ada dua cara penerapan yang dapat digunakan dalam memecahkan masalah Pentomino Puzzle dengan algoritma Depth First Search (selanjutnya disingkat DFS), yaitu :

1. DFS yang merujuk pada potongan pentomino
2. DFS yang merujuk pada sel kosong pada papan permainan.

#### 3.1.1 Merujuk pada Potongan Pentomino

Algoritma dengan cara ini adalah :

1. Mulai dengan papan permainan yang kosong.
2. Ambil pentomino pertama dan coba segala cara yang mungkin untuk meletakkannya.
3. Pada setiap cara tadi, lanjutkan dengan pentomino kedua dan coba kembali segala posisi yang mungkin untuk meletakkannya. Syarat penempatan pentominio adalah tidak keluar dari papan dan tidak menumpuk dengan pentomino sebelumnya.
4. Lakukan cara tersebut secara rekursif sehingga keduabelas pentomino berhasil menutupi papan permainan.

#### 3.1.2 Merujuk pada Papan Permainan

Algoritma dengan cara ini adalah :

1. Mulai dengan sel kosong pertama, yang berupa bujur sangkar, pada papan permainan.
2. Cobalah letakkan segala posisi dari semua pentomino yang memungkinkan.
3. Kemudian dari setiap langkah percobaan tersebut, lanjutkan dengan sel bujur sangkar yang belum tertutup. Coba kembali segala cara yang memungkinkan untuk menempatkan pentomino ke sel tersebut. Syarat penempatan pentomino adalah tidak keluar dari papan dan tidak menumpuk dengan pentomino sebelumnya.
4. Lakukan hal tersebut secara rekursif hingga seluruh bujur sangkar pada papan permainan tertutup dengan pentomino.

Kedua cara tersebut bersifat *exhaustive* tetapi pada akhirnya akan menghasilkan solusi yang sama. Namun, cara kedua lebih cepat dan mangkus jika dibandingkan cara yang pertama karena pada setiap langkah, lebih sedikit cabang pencarian yang dapat terbentuk. Pada umumnya, lebih banyak cara yang harus dilakukan untuk menempatkan pentomino tertentu pada papan permainan daripada untuk menutupi bujur sangkar tertentu dengan suatu pentomino. Dengan demikian, cara kedua membentuk pohon pencarian dengan *node* yang lebih sedikit.

## 4. Exact Cover Problem

Donald E. Knuth menunjukkan bahwa solusi penyelesaian Pentomino Puzzle dapat merujuk kepada masalah yang lebih umum, yang dikenal dengan *Exact Cover Problem*[1]. Dengan demikian, algoritma untuk memecahkan persoalan *Exact Cover Problem* dapat diterapkan juga dalam permainan Pentomino Puzzle. Hasilnya merupakan algoritma yang lebih mangkus dari algoritma lain yang telah dijabarkan di atas.

### 4.1 Penjelasan Exact Cover Problem

*Exact Cover Problem* adalah masalah pencarian *exact cover* atau 'penutup yang tepat' dalam suatu persoalan. Contoh dari permasalahan ini adalah sebagai berikut :

*Diberikan sebuah matriks 7 x 6 yang berisi 0 dan 1. Persoalannya adalah bagaimana memilih subset dari matriks tersebut yang jika digabungkan bersama dapat membentuk suatu matriks yang memiliki tepat satu angka 1 di setiap kolomnya.*

Contoh matriks :

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

Pada persoalan matriks tersebut, subsetnya adalah : baris 1, 4 dan 5.

### 4.2 Algoritma Exact Cover Problem

Algoritma untuk memecahkan Exact Cover Problem adalah :

1. Pilih suatu kolom *n* dari matriks
2. Pilih secara bergantian, setiap baris *m* dimana *n* memiliki satu angka 1. Untuk setiap *m* lakukan :
3. Hapus semua kolom dimana *m* mengandung satu angka 1 dan semua baris yang mengandung satu angka 1 pada kolom-kolom tersebut.
4. Lakukan algoritma ini berulang-ulang untuk mereduksi matriks dan memperoleh hasil.
5. Apabila ternyata tidak memperoleh solusi, lakukan runut balik langkah sebelumnya
6. Apabila telah ditemukan solusi, runut balik dn coba solusi lain

### 4.3 Penyelesaian Exact Cover Problem

Berikut ini adalah rangkaian langkah untuk menyelesaikan persoalan pada sub bab 4.1 :

Sebagai titik mula, diberi petunjuk pada matriks dan hilangkanlah semua angka 0 pada matriks hingga matriks menjadi :

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccccccc} A & B & C & D & E & F & G \end{array} \\ \begin{pmatrix} & & 1 & & 1 & 1 & \\ 1 & & & 1 & & & 1 \\ & 1 & 1 & & & 1 & \\ 1 & & & 1 & & & \\ & 1 & & & & & 1 \\ & & & 1 & 1 & & 1 \end{pmatrix} \end{array}$$

Langkah pertama, pilih kolom A. Kemudian, coba lanjutkan dengan baris 2.

Langkah ketiga membuat kita harus menghilangkan kolom A, D, dan G serta baris 2, 4, 5, dan 6. Sehingga yang tersisa adalah kolom B, C, E, dan F serta baris 1 dan 3.

$$\begin{array}{c} \\ 1 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} B & C & E & F \end{array} \\ \begin{pmatrix} & 1 & 1 & 1 \\ 1 & 1 & & 1 \end{pmatrix} \end{array}$$

Selanjutnya kita pilih kolom B dan baris 3. Dengan demikian maka kolom B, C, dan F serta baris 1 dan 3 pun hilang dan menyisakan kolom E tanpa baris.

*Matriks ini tidak memiliki solusi*. Maka kita harus melakukan runut-balik pilihan baris 3 pada kolom B. Karena tidak ada lagi angka 1 pada B yang dapat dipilih, maka dilakukan lagi runut-balik pilihan baris 2 pada kolom A.

Kemudian dipilih baris 4 kolom A. Sekarang dihilangkan kolom A dan D serta baris 2, 4 dan 6. Hasilnya adalah :

$$\begin{array}{c} \\ 1 \\ 3 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} B & C & E & F & G \end{array} \\ \begin{pmatrix} & 1 & 1 & 1 & \\ 1 & 1 & & 1 & \\ 1 & & & & 1 \end{pmatrix} \end{array}$$

Selanjutnya kita pilih kolom B dan baris 3 lagi. Dengan demikian maka kolom B, C, dan F serta baris 1, 3, dan 5 pun dihilangkan. Sekali lagi berakhir pada kesimpulan kolom E tanpa baris. Maka sekali kagi kita merunut balik dan pilih baris 5 bukan baris 3. Kemudian kolom B dan G serta baris 3 dan 5 pun dihilangkan, dan meninggalkan :

$$\begin{array}{c} \\ 1 \end{array} \begin{array}{c} \begin{array}{ccc} C & E & F \end{array} \\ \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \end{array}$$

Kemudian terakhir pilih baris 1 dan menghilangkan seluruh baris dan kolom. Hal tersebut berarti berhasil. Pemecahan dari persoalan adalah baris-baris yang kita pilih, yaitu : 4, 5 dan 1.

Agar banyaknya cabang semakin kecil pada tahap 2 algoritma, maka pada tahap pertama pilih kolom matriks yang memiliki paling sedikit angka 1.

Dari langkah-langkah solusi di atas dapat dilihat di sini bahwa pemecahan persoalan *Exact Cover problem* menggunakan algoritma DFS dan algoritma *backtracking* (runut-balik). Hal ini dapat juga digunakan untuk memecahkan persoalan Pentomino Puzzle.

## 5. Analisis Penyelesaian Masalah Pentomino Puzzle

Pada bab ini akan dijabarkan analisis penerapan algoritma *Exact Cover Problem* untuk menyelesaikan persoalan yang telah dikemukakan pada bab 3.

Untuk menerapkan algoritma yang serupa dengan persoalan *Exact Cover Problem* pada penyelesaian persoalan Pentomino Puzzle, bentuk matriks yang terdiri dari 72 kolom. 12 pertama mewakili tiap bentuk pentomino. Dan 60 mewakili tiap ruang bujur sangkar pada papan permainan.

Sekarang, setiap cara sebuah pentomino dapat diletakkan pada papan, tambahkan baris pada matriks. Baris tersebut harus mengandung sebuah angka 1 pada salah satu dari 12 kolom yang menggambarkan bentuk pentomino yang digunakan. Kemudian lima angka 1 pada 60 kolom berikutnya, sesuai dengan posisi pentomino tersebut pada papan. Yang lainnya berisis angka 0.

Hal tersebut dilakukan pada semua pentomino dalam berbagai posisi. Pada persoalan bujur sangkar 8 x 8 dengan lubang bujur sangkar 2 x 2 diatas, terdapat 1568 baris atau 1568 kemungkinan posisi dari pentomino-pentomino.

Langkah selanjutnya untuk menyelesaikan persoalan adalah dengan menggunakan algoritma penyelesaian Exact Cover problem. Jika matriks dapat habis semua, maka telah ditemukan solusi peletakkan pentomino-pentomino secara tepat.

Untuk lebih jelasnya adalah sebagai berikut :

**Membangkitkan matriks Pentomino Puzzle**

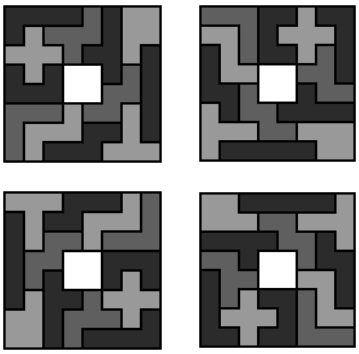
1. Bentuk matriks dengan kolom sebanyak 72 buah yang 12 pertama mewakili tiap bentuk potongan pentomino dan 60 selanjutnya adalah tiap ruang bujur sangkar pada papan permainan.
2. Letakkan angka 1 pada salah satu bagian pentomino dan angka 1 sebanyak lima pada posisi setiap bujur sangkar yang tertutup oleh pentomino tersebut.
3. Angkat pentomino tersebut dan anggap papan kosong kembali.
4. Lakukan berulang kali untuk semua posisi yang memungkinkan
5. Lakukan berulang kali untuk seluruh pentomino

**Memecahkan Pentomino Puzzle**

Sama dengan pemecahan *Exact Cover Problem*, langkahnya adalah

1. Pilih suatu kolom *n* dari matriks
2. Pilih secara bergantian, setiap baris *m* dimana *n* memiliki satu angka 1. Untuk setiap *m* lakukan :
3. Hapus semua kolom dimana *m* mengandung satu angka 1 dan semua baris yang mengandung satu angka 1 pada kolom-kolom tersebut.
4. Lakukan algoritma ini berulang-ulang untuk mereduksi matriks dan memperoleh hasil akhir matriks yang kosong.
5. Apabila ternyata tidak memperoleh solusi, lakukan runut balik langkah sebelumnya
6. Apabila telah ditemukan solusi, runut balik dn coba solusi lain (solusi bisa lebih dari satu)

Dari persoalan bujur sangkar 8 x 8 dengan lubang bujur sangkar 2 x 2 diatas diperoleh solusi sebanyak 65 buah. Dimana sebenarnya 520 solusi namun harus dibagi empat karena satu solusi dapat diputar empat kali. Antara lain sebagai berikut :



Keempat solusi tersebut sesungguhnya adalah sama namun diputar pada ke-empat sisinya.

## 6. Kesimpulan

Dalam menyelesaian permasalahan Pentomino Puzzle terdapat berbagai algoritma yang dapat digunakan. Antara lain algoritma Depth First Search (DFS) serta algoritma yang menyelesaikan *Exact Cover Problem* yang merupakan gabungan dari algoritma Depth First Search (DFS) dan algoritma *backtracking*. Hingga saat ini algoritma *Exact Cover Problem* yang diutarakan oleh Donald E. Knuth merupakan salah satu solusi yang cukup mangkus untuk menyelesaikan Pentomino Puzzle.

Adapun yang kadang menjadi permasalahan adalah pendirian matriks dari persoalan yang harus dipecahkan. Namun, jika algoritma pendirian

matriks telah berjalan dengan baik, maka permasalahan Pentomino Puzzle dapat terselesaikan dengan baik.

## 7. Daftar Pustaka


[1] Knuth, Donald E. 1973. "*The Art Of Computer Programming, vol. 3: Sorting And Searching*". Addison Wesley.

[2] Solving the pentomino problem http://svana.org/kleptog/puzzles/pentomino.html . Diakses pada tanggal 19 Mei 2005 pukul 19.30 WIB

[3] Pentomino http://mathworld.wolfram.com/Pentomino.html Diakses pada tanggal 19 Mei 2005 pukul 19.30 WIB