Penerapan Algoritma Knuth X pada Proses Pemecahan Masalah Logic Puzzle

Steve Immanuel Harnadi / 13512035
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
steve.harnadi@gmail.com, 13512035@std.stei.itb.ac.id

***Abstraksi*—Logic Puzzle pada dunia game seperti Sudoku, Tetris, ataupun Kanoddle merupakan permainan puzzle termasuk. Meskipun puzzle-puzzle tersebut kelihatannya membosankan dan hanya cocok dimainkan oleh anak-anak namun di dalamnya terdapat esensi strategi pemecahan yang luar biasa menantang kita. Seorang ilmuwan bernama Daniel Knuth merancang strategi khusus untuk memecahkan masalah puzzle seperti ini secara optimal, dengan pendekatan exact cover problem. Pendekatan tersebut diterapkan menjadi algoritma Knuth X yang menggunakan konsep dancing link dalam implementasinya. Algoritma ini secara khusus menerapkan strategi backtracking, iteratif, dan rekursif sekaligus sampai solusi puzzle terpecahkan secara optimal.**

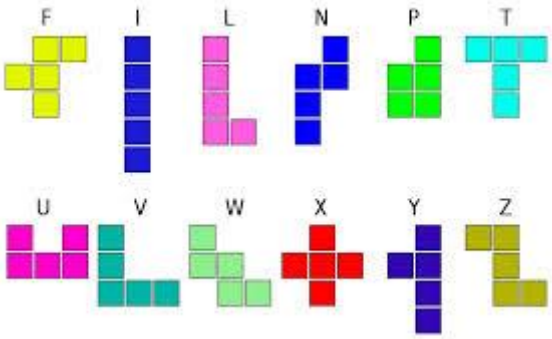
***Kata Kunci*—Backtracking, *Dancing Links*, Exact Cover Problem, Logic Puzzle**

I. PENDAHULUAN

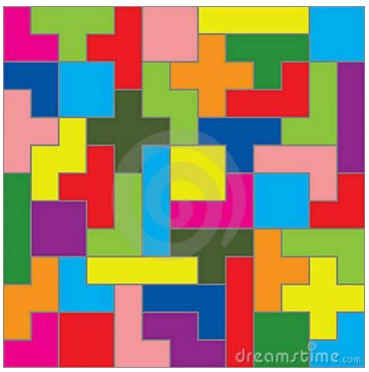
Pada era teknologi modern saat ini, permainan game semakin marak dan kreatif di industri gadget. Berbagai aplikasi permainan pada platform android mulai membanjiri gadget dengan berbagai fitur-fitur menarik. Aplikasi game yang ada bukan saja melatih kreativitas pemainnya, namun juga ada yang mengasah kemampuan berpikir kita. Hanya saja dengan terbatasnya waktu yang kita miliki, sebenarnya terdapat game-game tertentu yang dapat kita teliti dan eksplorasi lebih lanjut lagi, namun belum tersentuh oleh gadget kita.

Kebanyakan dari kita biasanya malas memainkan game yang menuntut kita untuk sedikit berpikir cerdas. Sebagian besar mahasiswa sekali pun lebih senang dengan game yang praktis berbau skill dan keberuntungan semata. Namun, game-game cerdas semacam itulah yang sarat dengan strategi-strategi pemecahan yang menarik yang patut untuk ditelaah lebih lanjut, sebab di dalamnya terdapat teka-teki menarik yang sebenarnya terdiri dari ratusan bahkan ribuan solusi yang berbeda-beda tergantung dari seberapa efektif kita mencari solusinya.

Ambil contoh game tetris yang merupakan permainan menyusun blok-blok yang terdiri dari berbagai bentuk potongan balok dan variasi dimensi (2D dan 3D). Varian game ini pun bermacam-macam, namun sebenarnya intinya sama. Pemecahan masalah game ini meskipun sukar diprediksi namun dapat diusahakan agar penyusunan potongan balok-balok tersebut seoptimal mungkin. Implementasinya dapat menggunakan algoritma yang menggunakan pendekatan Exact Cover Problem yaitu algoritma Knuth X (akan diuraikan kemudian). Berikut adalah contoh potongan balok tetris yang harus disusun menjadi rangkaian balok yang sekontigu mungkin seperti pada ilustrasi berikut.



Gambar 1. Potongan Tetris yang Biasa Digunakan Dalam Permainan Game.
Sumber:http://euler.slu.edu/escher/index.php/Slides,_Flips_and_Turns



Gambar 2 : Balok-balok Tetris di atas disusun sedemikian rupa agar terbentuk bangun yang sekontigu mungkin (sedikit lubang / celah kosong).

Sumber : http://www.dreamstime.com/royalty-free-stock-image-tetris-game-pieces-image13362826

Contoh game lainnya adalah game Sudoku yang sudah tidak asing lagi di telinga kita saat ini. Permainan ini merupakan permainan kombinatorik angka di mana kita ditugaskan menulis angka 1-9 pada masing-masing lajur baris dan kolom pada petak berukuran 9 x 9 tanpa tiap lajur dan kolom dari petak tersebut mengandung angka 1-9 lebih dari 1 (harus terisi 1-9). Game ini menuntut kita agar efisien dalam penyusunan angka sehingga tidak boros waktu dan tenaga. Berikut adalah contoh gambar game

Sudoku yang sudah banyak dikenal tersebut.

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

Gambar 3 : Permainan Sudoku yang menuntut pemain untuk mengisi angka di kotak 9 x 9 seefisien mungkin. Sumber : http://webdocs.cs.ualberta.ca/~smillie/APE/APE50.html

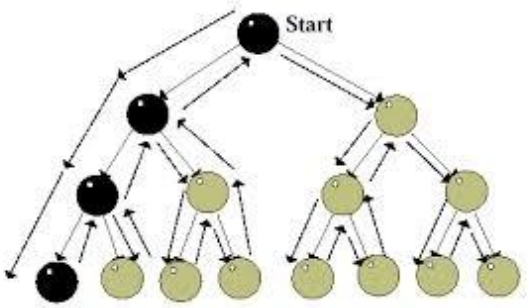
Atas dasar pemikiran itulah penulis mencoba mengembangkan ide seputar game-game logic puzzle yang di dalamnya terdapat gagasan-gagasan menarik seputar proses pemecahannya. Di antara sekian banyak metode, penulis mencoba menggali metode algoritma Knuth X yang menggunakan pendekatan Dancing Links yang memakai konsep Exact Cover Problem. Alasannya, proses pemecahannya berbeda dengan strategi algoritma lainnya, di mana pendekatan ini menggunakan strategi optimasi dan dinamis sekaligus.

II. DASAR TEORI

2.1 Prinsip Algoritma Backtracking :

Salah satu pendekatan strategi algoritma yang cukup populer dan setidaknya memperbaiki kinerja pendekatan strategi brute force adalah strategi backtracking. Strategi ini menjadikan pohon ruang status (himpunan) solusi yang mungkin sebagai acuan pencarian solusi. Adapun penelusuran solusi menggunakan metode DFS (Depth First Search), di mana pencarian solusi dilakukan sampai ke akar pohon ruang status yang terdalam terlebih dahulu untuk kemudian dilakukan backtrack pencarian ke simpul akar di atasnya secara rekursif.

Berikut adalah ilustrasi pencarian solusi secara DFS di mana simpul-simpul pada pohon ruang status ditelusuri sampai ke akar terdalam terlebih dahulu.



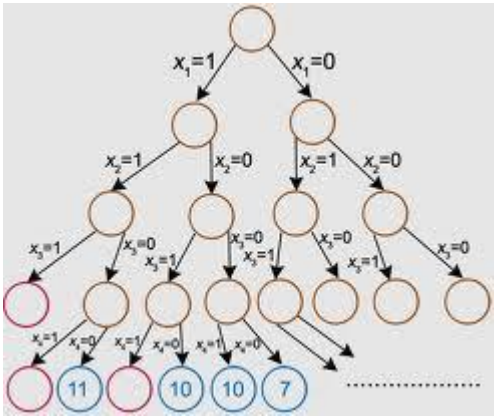
Gambar 4 : Penelusuran pencarian solusi secara DFS pada pohon ruang statu. Sumber : https://www.cs.drexel.edu/~introcs/F2K/lectures/7.2_AI/Formal3.html

Backtracking merupakan strategi algoritma yang menerapkan pencarian solusi secara DFS. Pada strategi algoritma Backtracking, simpul solusi (simpul pada pohon ruang status) dilahirkan satu per satu dengan membentuk lintasan dari akar ke daun. Salah satu komponen terpenting dari strategi ini adalah adanya fungsi pembatas (dijelaskan di bawah) yang membatasi pembentukan simpul jika pembentukan simpul tersebut sudah tidak memungkinkan lagi mengarah ke solusi. Jika penelusuran solusi sudah sampai di simpul daun, maka penelusuran dilanjutkan ke cabang berikutnya dari simpul akar terdekat, demikian seterusnya.

Adapun unsur-unsur utama yang harus ada pada strategi pencarian secara backtracking adalah sbb.

1. **Solusi Persoalan**
   Solusi dinyatakan sebagai vector dengan n-tuple yang dinyatakan sebagai :
   $X = (x_1, x_2, \ldots, x_n)$
   Dengan x_i elemen himpunan berhingga Si.
2. **Fungsi Pembangkit nilai x_i**
   Fungsi pembangkit ini digunakan untuk membangkitkan komponen vektor solusi x_i (dinyatakan sebagai T(i)).
3. **Fungsi Pembatas**
   Fungsi ini merupakan fungsi yang menentukan apakah komponen vektor solusi X mengarah ke solusi yang diinginkan. Jika tidak maka solusi terkini akan dibuang dan tidak dipertimbangkan lagi. Jika ya, pembangkit nilai melanjutkan pembangkitan komponen vektor solusi X berikutnya.

Berikut adalah contoh pohon ruang status hasil penelusuran dengan strategi backtracking :



Gambar 5 : Contoh pohon ruang status penerapan backtracking pada masalah N-Queen Problem. Sumber : http://www.loria.fr/~quinson/blog/2011/0816/JLM_progress/

2.2 Pengenalan Pendekatan Exact Cover Problem :

Exact Cover Problem merupakan salah satu masalah optimasi NP-problem yang sering digunakan untuk mencari solusi optimal dari serangkaian kemungkinan solusi yang bisa berubah-ubah tergantung kondisi dari luar. Pada kasus ini, kondisi yang mempengaruhi serangkaian

solusi adalah batas petak (pada permainan tetris) atau batas angka (pada permainan Sudoku). Exact Cover Problem bisa juga disebut sebagai "Pendekatan Trial dan Error yang Pasti", karena pemecahan masalahnya sebenarnya tidak terlalu sulit (hanya membutuhkan runut balik rekursif).

Komponen yang mutlak terdapat pada Exact Cover Problem adalah semua himpunan kandidat solusi yang mungkin, di mana masing-masing himpunan memuat elemen-elemen yang dapat mengisi constrain secara optimum. Kemudian, himpunan tersebut dituangkan dalam matriks N x M, di mana lajur baris menyatakan semua himpunan kandidat solusi yang mungkin dan constrain yang terkait dengan masing-masing kandidat tersebut digambarkan dalam lajur kolom ke kanan (pada mulanya kita harus mendefinisikan dahulu himpunan kandidat yang ada sebelum menuangkannya dalam matriks). Langkah berikutnya adalah mengumpulkan semua himpunan kandidat (lajur baris) yang tepat men-cover kolom yang hanya mempunyai satu nilai saja. Adapun semua himpunan kandidat yang tidak mungkin mempunyai solusi tidak mempunyai constrain value yang bertepatan dengannya (dikosongkan saja).

	Pants Pocket?	Coat Pocket?	Backpack?
Entries			
Keys in pants	1	0	0
Keys in coat	0	1	0
Keys in backpack	0	0	1
Wallet in pants	1	0	0
Wallet in coat	0	1	0
Wallet in backpack	0	0	1
Phone in pants	1	0	0
Phone in coat	0	1	0
Phone in backpack	0	0	1

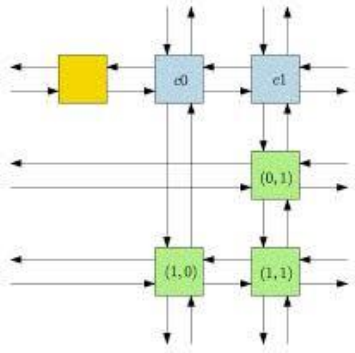
Gambar 6 : Contoh sederhana representasi matriks exact cover problem yang ditandai dengan Boolean value. Kolom paling kiri menandakan lajur baris (kemungkinan solusi) dan ketiga lajur kolom menandakan kemungkinan constrain yang ada.
Sumber:http://gieseanw.wordpress.com/2011/06/16/solving-sudoku-revisited/

Langkah berikutnya yang dilakukan adalah menandai (menghapus) lajur kolom yang hanya memuat satu kandidat. Penandaan dilakukan oleh suatu baris khusus yang membandingkan konstrain terkait dengan kandidat solusi yang mungkin. Proses "cover" berhenti kala pembacaan yang dilakukan oleh baris ini semuanya sudah ditandai (dalam beberapa literatur pembacaan selesai dilakukan ketika semua baris hanya terdiri dari angka 1 saja). Solusi diperoleh dari elemen lajur kolom yang dihapus dari penelusuran sebelumnya.

Inti dari exact cover problem secara garis besar (dari yang diuraikan sebelumnya) adalah mencari dari sekian banyak pilihan solusi yang mungkin (diberikan himpunan konstrain tertentu yang disediakan), solusi yang memenuhi setiap konstrain masing-masing satu kali saja. Permasalahan ini jelas cocok dengan masalah logic puzzle umumnya yang akan dipecahkan kemudian di bagian analisis.

2.3 Pengenalan Metode Dancing Links :

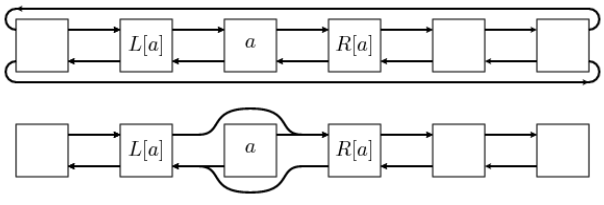
Secara garis besar, metode Dancing Links merupakan metode pemecahan masalah kompleks yang melibatkan struktur data yang besar dan linked list yang tersusun secara dinamis. Metode ini cocok digunakan untuk menyelesaikan permasalahan exact cover problems. Metode ini dilakukan secara rekursif (berulang) namun juga backtracking, sampai seluruh node yang tersedia dikeluarkan dari linked list yang tersedia Adapun ilustrasi abstrak dari pemecahan masalah secara dancing links dapat dijelaskan dengan gambar di bawah ini. Diketahui node inisialisasi merupakan node berwarna kuning.



Gambar 7 : Salah satu contoh dancing links yang terdiri node-node yang memiliki link double dua arah di setiap sisinya. Lintasan link membentuk pola sirkular tertentu
Sumber : http://kunigami.wordpress.com/2013/04/28/the-algorithm-x-and-the-dancing-links/

Elemen-elemen dancing links adalah node inisialisasi di sisi kiri dan node-node yang memiliki pointer hingga ke dua hingga empat arah tergantung pada posisi node tersebut dalam petak-petak yang tersedia. Adapun garis-garis yang menghubungkan antar node membentuk lintasan sirkular tertentu yang mempunyai pola. Node yang merupakan bagian dari link dapat diexclude dari rangkaian link dengan memutus pointer linked list sebelum dan sesudah node tersebut. Jika node tersebut ingin dikembalikan lagi ke dalam linked list, maka yang perlu dilakukan hanyalah menghubungkan kembali garis linked list ke node tersebut (mencatat kembali pointer ke node tersebut) dan dari node tersebut ke node berikutnya.

Ilustrasi pemutusan node pada dancing links dapat diibaratkan sebagai berikut. Perlu diketahui juga bahwa setiap node mencatat setiap elemen/pointer dari node-node yang bersebelahan dengan dirinya.



Gambar 8 : Ilustrasi dancing links sederhana yang

terdiri dari 1 baris beserta pemutusan elemen node pada linked list tersebut. Keterangan : L[a] = nilai node sebelah kiri a, R[a] = nilai node sebelah kanan a. Nilai node di sebelah kiri dan kanan dicatat karena setiap node mempunyai link double yang bersifat dua arah sekaligus (bisa 4 arah tergantung permasalahan). Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle

III. ANALISIS PEMECAHAN MASALAH

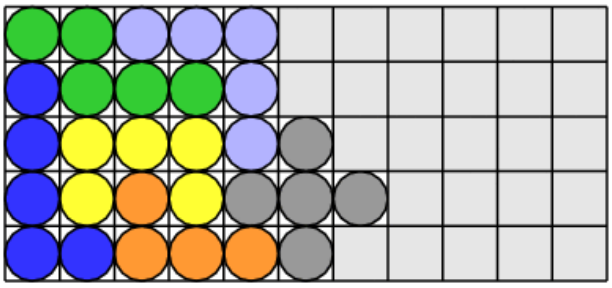
3.1 Implementasi Algoritma Knuth X dalam Proses Pemecahan Logic Puzzle :

Dalam pembahasan bab ini, penulis mengambil dua contoh logic puzzle yang sudah pernah disinggung sebelumnya, yaitu tetris dan sudoku dalam pembuktian implementasi algoritma Knuth X. Inti kedua game tersebut adalah sama, yaitu bagaimana setiap kandidat solusi menutupi tepat satu konstrain yang berlaku. Pada game Tetris target utama yang hendak dicapai : mencari potongan balok/pola yang tepat mencover 1 satuan petak pada bidang 2D/3D yang tersedia. Sedangkan pada sudoku : mencari angka pada 1 satuan kotak (dari bidang kotak 9 x 9) yang tepat hanya muncul 1 kali pada kolom dan baris terkait. Adapun algoritma Knuth X sendiri memanfaatkan metode dancing links yang dipakai dalam permasalahan exact cover problem dalam proses penyelesaiannya.

Implementasi Exact Cover Problem pada game tetris 2D (dengan ukuran bidang 5 x 11 petak) akan dijelaskan pada uraian di bawah ini.

1. Penentuan Himpunan Kandidat Solusi :
   Misalkan terdapat potongan-potongan pola yang ingin dimasukkan ke dalam grid (bidang) 5 x 11 petak seperti pada gambar berikut ini.

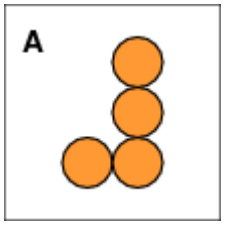
(a)



(b)

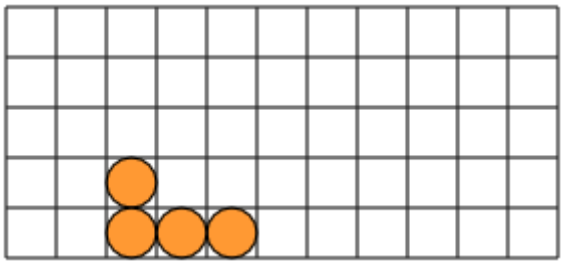
Gambar 9 : Ilustrasi permainan susun kanoddle yang terdiri dari potongan-potongan pola dengan 12 jenis (gambar (a)) yang disusun dalam bidang berukuran 5 x 11 petak (gambar (b)) Setiap pola pada gambar a) memperlihatkan kandidat solusi yang berbeda. Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle

maka setiap potongan pola di atas masing-masing dapat dideskripsikan menjadi himpunan solusi dengan parameter yang berbeda-beda tergantung pada posisi mana pola tersebut diletakkan di dalam grid. Sebagai contoh, perhatikan pola berikut ini.



Gambar 10 : Pola Kanoddle dengan bentuk L. Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle

Pola pada gambar di samping kemudian dimasukan ke dalam grid dengan ukuran 5 x 11 petak seperti gambar di bawahnya. Maka, himpunan kandidat solusi terdiri dari empat bagian (empat petak yang diisi oleh pola ini) di mana masing-masing bagian berisi koordinat grid pada masing-masing bagian pola yang mengisi satu petak. Jika pola di atas diisikan ke grid 5 x 11 dengan posisi sebagai berikut, maka himpunan solusi terdefinisi adalah A = [4,3], [5,3], [5,4], [5,5] untuk potongan pola tersebut. Perhatikan gambar pola L di bawah ini.



Gambar 11 : Pola kanoddle gambar 10 diisikan secara sembarang pada grid 5 x 11 petak. Pola mengandung empat buah titik yang terletak pada petak yang berbeda menjadi status dari pola itu sendiri. Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle

2. Konversi Himpunan Solusi ke Dalam Bentuk Matriks Tereduksi :
   Langkah berikutnya adalah mengkonversi setiap himpunan solusi yang didapatkan dari setiap pola ke dalam bentuk matriks tereduksi (dengan pola true/false atau 1/0 (lihat bagian dasar teori)). Ambil contoh pola pada gambar 11. Jika dituangkan ke dalam bentuk matriks seperti yang sudah dipaparkan pada bab prinsip dasar, maka himpunan solusi dapat dikonversikan menjadi matriks pada tabel di bawah ini. Matriks memiliki lajur baris yang merupakan

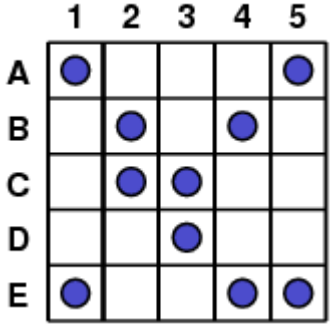
banyaknya himpunan kandidat solusi dan lajur kolom merupakan seluruh kemungkinan nilai elemen dari himpunan vektor solusi (dalam hal ini, nilai elemen maksimal 5, karena itu lajur kolom yang dibentuk sebanyak 5 buah). Selanjutnya, setiap elemen matriks diisi nilai 1 yang berarti himpunan lajur baris memiliki elemen pada lajur kolom, atau nilai 0 yang berarti himpunan terkait tidak mengandung nilai pada lajur kolom tersebut.

	1	2	3	4	5
A	0	0	1	1	0
B	0	0	1	0	1
C	0	0	0	1	1
D	0	0	0	0	1

Tabel 1 : Contoh konversi matriks reduksi dari pola pada gambar 10.

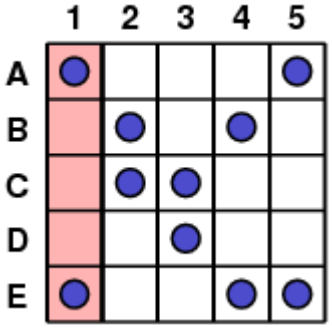
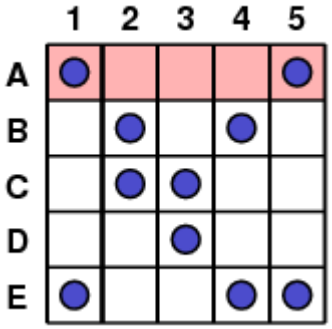
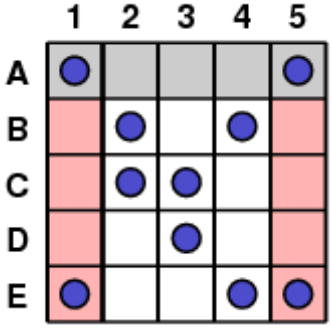
3. Eliminasi baris dan kolom pada matriks tereduksi :
   Langkah terakhir yang perlu dilakukan adalah mereduksi matriks yang sudah dihasilkan dari himpunan kandidat solusi sehingga dihasilkan matriks akhir yang hanya mengandung elemen 0 (berarti bukan solusi) atau setiap lajur kolom hanya mempunyai tepat satu buah elemen 1 saja (merupakan solusi akhir). Untuk lebih jelasnya, berikut diuraikan algoritma deskriptif terkait proses reduksi matriks.
   a) Jika matriks tidak memiliki kolom sedikitpun (sudah habis), maka solusi ditemukan. STOP.
   b) Jika terdapat kolom tidak memiliki elemen 1, maka solusi tak ditemukan. STOP.
   c) Pilih sebuah kolom c.
   d) Perhatikan baris-baris r yang berkorelasi dengan kolom c tersebut. Kemudian, pada setiap baris r tersebut, perhatikan juga kolom lain (misalkan c') yang berkorelasi dengan baris r terkait. Hapus baris dan kolom yang saling berkorelasi tersebut. Keterangan : berkorelasi = kolom dan baris bersilangan/beririsan pada elemen 1.
   e) Ulangi langkah a) untuk matriks baru yang mengalami proses reduksi tersebut.

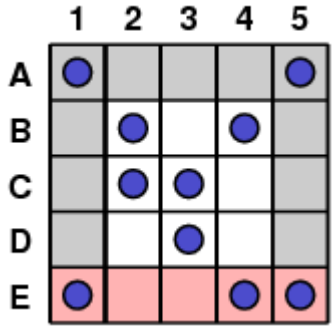
   Jadi, matriks pada tabel 1 di atas bukan merupakan solusi karena terdapat kolom yang semua elemennya false (bernilai 0). Hal tersebut memang sesungguhnya jelas karena memang pola tersebut belum membentuk solusi logic puzzle Kanoddle yang sesungguhnya. Untuk memperjelas reduksi matriks, ambil contoh matriks reduksi pada gambar di bawah ini.



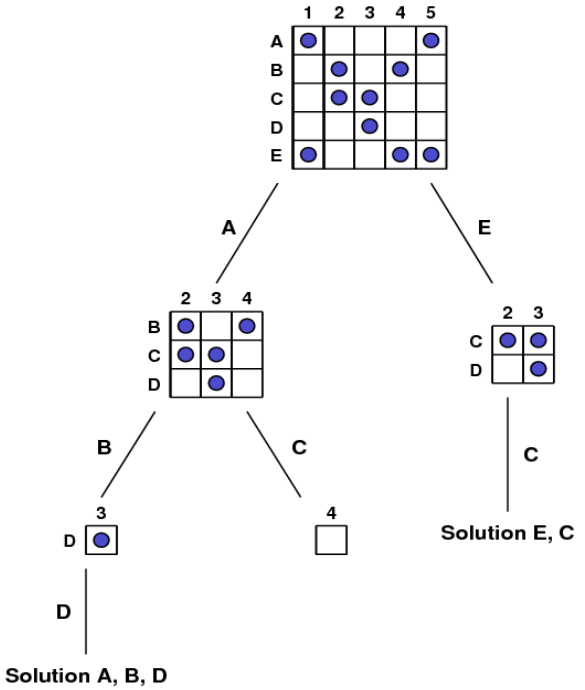
Gambar 12 : Contoh matriks reduksi salah satu exact cover problem.
Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle

Langkah-langkah untuk mereduksi matriks pada gambar 12 di atas lebih lanjut dijelaskan dalam tabel berikut ini.

PROSES	PENJELASAN
 *Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle*	Ambil kolom 1 sebagai pilihan. Perhatikan baris yang berkorelasi dengan kolom 1, yaitu baris A (himpunan solusi A) dan baris E (himpunan solusi E). Kedua baris ini berikutnya dibedakan status matriksnya.
 *Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle*	Pada baris A, perhatikan lajur kolom yang terkandung ke dalam himpunan solusi A tersebut. Lajur kolom yang berkorelasi dengan baris A (kolom 1 dan 5) ditandai.
 *Sumber : http://www.ams.org/samplings*	Pada kolom 5 yang berkorelasi dengan baris A, expand baris yang berkaitan dengan kolom 5 tersebut, apakah kolom tersebut mengandung lebih dari 1 buah nilai true. Lanjutkan ekspansi baris dari baris yang

/feature-column/fcarc-kanoodle	berkorelasi dengan kolom tersebut.
 *Sumber : http://www.ams.org/samplings /feature-column/fcarc-kanoodle*	Pada status akhir, terbentuk blok area yang terdiri dari baris dan kolom, di mana setiap baris dan kolom di blok tersebut mengandung lebih dari 1 nilai true. Kemudian hapus blok tersebut dari matriks menghasilkan matriks baru. Ulangi proses dari awal untuk baris E (menghasilkan matriks baru yang berbeda dari matriks dari baris A).

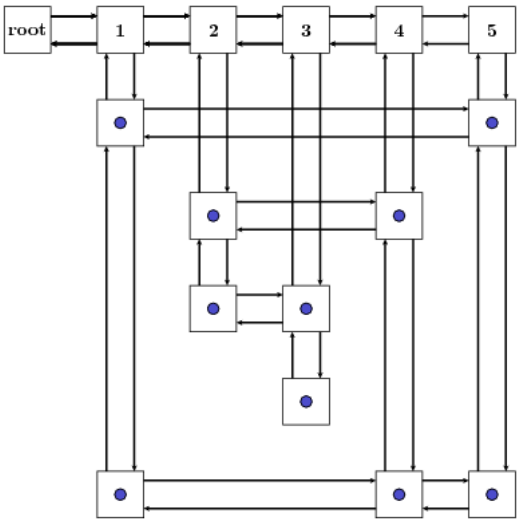
Hasil akhir dari matriks reduksi di atas menghasilkan struktur pohon (yang nantinya diterapkan secara backtracking pada struktur dancing links) pada gambar di bawah ini. Struktur pohon inilah yang menjadi dasar pengerjaan dancing links selanjutnya yang diterapkan pada penempatan pola Kanoddle.



Gambar 13 : Struktur pohon hasil exact cover reduction yang terbentuk dari matriks reduksi gambar 12. Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle

Proses reduksi matriks pada pembahasan kemudian digunakan pada metode dancing links sebagai acuan utama. Tinjau kembali logic puzzle Kanoddle sebagai permasalahan menempatkan pola dengan bentuk berbeda ke dalam grid 5 x 11 petak. Implementasi algorithma Knuth X pada permainan ini adalah membentuk 2 matriks reduksi raksasa dengan setiap elemen yang bernilai 1/true menandakan satu petak tepat terpakai 1 kali oleh pola tertentu. Lajur baris pada matriks tersebut merupakan himpunan pola yang membentuk solusi (jadi terdapat 12 kandidat solusi, lihat gambar 9) sedangkan lajur kolom dibedakan berdasarkan constrain kolom dan baris pada grid (matriks reduksi 1 mengandung kolom sebanyak 5 jumlah baris pada grid, matriks reduksi 2 sebanyak 11 jumlah kolom pada grid). Solusi puzzle Kanoddle ditemukan apabila setiap pola digunakan masing-masing 1 kali dan setiap petak hanya digunakan sebanyak 1 kali tanpa ada pola yang saling bertindihan (setiap matriks pada lajur kolomnya hanya mengandung 1 buah elemen yang bernilai true/1 (menandakan setiap konstrain tepat dipakai oleh satu buah kandidat solusi saja)).

Berikut adalah contoh struktur dancing links yang dapat memperjelas pemahaman kita dalam menerapkan algoritma Knuth X.



Gambar 14 : Salah satu contoh struktur Dancing Links pada penerapan algoritma Knuth X dari matriks reduksi pada gambar 12. Sumber : http://www.ams.org/samplings/feature-column/fcarc-kanoodle

Elemen-elemen pada struktur dancing links di atas dapat diexclude dan diinclude kembali seperti pada ilustrasi gambar 8. Adapun status struktur dancing links dapat berubah-ubah membentuk pohon ruang status seperti pada gambar 13, di mana penelusuran pohon status seperti ini dilakukan secara backtracking dengan fungsi pembangkit dan fungsi pembatas adalah aturan pada metode exact cover problem seperti yang sudah dituangkan dalam

algoritma exact cover pada halaman sebelumnya. Proses ini berlangsung sampai tidak ada simpul dari pohon ruang status tersebut yang tidak dapat diexpand lebih lanjut ke arah solusi.

Adapun permasalahan logic puzzle lainnya seperti tetris dan sudoku juga dapat diselesaikan dengan implementasi algoritma Knuth X seperti yang sudah dijelaskan pada logic puzzle Kanoddle. Perbedaannya hanyalah terletak pada himpunan kandidat solusi dan constrain kolom yang terdefinisi pada matriks reduksi yang dibentuk. Pada permainan tetris, terdapat tiga konstrain yang harus diberlakukan karena grid terdiri dari ruang 3 dimensi di mana masing-masing unit grid harus terpakai hanya oleh 1 komponen pola saja. Sedangkan pada permainan Sudoku, konstrain yang berlaku adalah jumlah kolom dan tabel sebanyak 9 buah (dijadikan sebagai lajur kolom pada matriks reduksi). Hanya saja, karena pada Sudoku terdapat 9 kotak kecil lainnya di dalam grid utama, maka dancing links pada permainan Sudoku diterapkan secara iteratif dimulai dari kotak terkecil terlebih dahulu, kemudian berlanjut pada kotak yang besar.

3.2 Analisis Algoritma Knuth X :

Bagian ini akan menguraikan keuntungan dan juga kerugian penerapan algoritma Knuth X pada permasalahan logic puzzle. Di samping itu, diuraikan juga tingkat kompleksitas penyelesaian algoritma Knuth X jika dibangdingkan dengan algoritma brute force pada umumnya.

Algoritma Knuth X merupakan metode pemecahan logic puzzle yang cukup praktis, sebab kita tidak perlu mencoba semua kemungkinan solusi satu per satu seperti pada metode brute force. Jika kita mengenumerasikan semua kemungkinan solusi kemudian mengeceknya satu per satu seperti pada penelusuran exhaustive search, maka pada puzzle Kanoddle yang sudah diuraikan sebelumnya terdapat (4 x 12)! kemungkinan solusi yang sangat tidak efisien (kompleksitas : O(n!)). Begitu pula dengan permainan mengisi angka Sudoku 9 x 9 membutuhkan enumerasi solusi sebanyak maksimal 81 x 9 kemungkinan yang juga tidak efisien (kompleksitas : $O(n^6)$). Dengan algoritma Knuth X yang menggunakan pendekatan backtracking, semua kemungkinan solusi yang sudah tidak mungkin mengarah ke jawaban tidak akan dipertimbangkan lagi sehingga membuat kemungkinan solusi yang harus diperiksa menjadi lebih sedikit, sebab terdapat fungsi pembatas yang membunuh komponen solusi yang tidak mungkin mengarah ke solusi.

Algoritma Knuth X sendiri sebenarnya merupakan pengembangan mutakhir dari strategi backtracking yang dipadukan dalam struktur data yang dinamis. Jika pemecahan logic puzzle dilakukan secara backtracking, sesungguhnya proses mencari pohon ruang solusi juga harus mempertimbangkan berapa banyak simpul yang harus dibangkitkan. Artinya, semua pohon ruang status yang utuh dari puzzle harus diketahui terlebih dahulu sebelum penelusuran pohon dilakukan secara DFS. Pada algoritma Knuth X, prinsipnya sebenarnya juga adalah DFS, hanya saja kita dapat langsung menerapkannya tanpa perlu mengetahui ruang solusi secara keseluruhan.

Kelemahan algoritma Knuth X adalah ketidakmampuan untuk menyelesaikan permasalahan logic puzzle yang bersifat unpredictable (artinya tergolong ke dalam NP problems). Proses penelusuran yang sudah dipaparkan sebelumnya dilakukan dengan asumsi bahwa potongan pola yang tersedia sudah terdefinisi (sudah pasti, tidak berubah-ubah lagi, dan pasti menghasilkan solusi akhir). Sedangkan game tetris lebih bersifat NP problems di dunia nyata ketimbang P problems.

IV. KESIMPULAN

Kesimpulan akhir dari makalah ini adalah penyelesaian masalah logic puzzle dengan algoritma Knuth X patut dicoba pada penyelesaian masalah logic puzzle seperti menyusun balok atau angka yang memenuhi konstrain tertentu. Hanya saja, penggunaannya juga terbatas pada game yang bersifat P problem saja. Meskipun demikian, algoritma ini merupakan algoritma yang mutakhir yang dapat diterapkan dalam waktu yang singkat dengan struktur data dancing link yang fleksibel.

REFERENSI

[1] http://www.ams.org/samplings/feature-column/fcarc-kanoodle (Tanggal Akses : 15 Mei 2014, pukul 10.30)

[2] Munir, Rinaldi. 2012. *Diktat IF2211 : Strategi Algoritma*. Bandung : Teknik Informatika ITB.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 15 Mei 2014

Steve Immanuel Harnadi
13512035