Penyelesaian Permasalahan *Nonogram* dengan Algoritma Runut Balik

Hendra Hadhil Choiri (135 08 041)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
hendra_h2c@students.itb.ac.id

***Abstract*—Permainan nonogram memang cukup jarang diketahui orang, karena memang permasalahan ini cukup kompleks, bahkan sudah dibuktikan ini merupakan masalah NP-complete. Sebenarnya permainan ini cukup menarik, yakni mengatur piksel membentuk suatu gambar berdasarkan petunjuk angka-angka yang diberikan. Ada banyak metode yang bisa digunakan untuk menyelesaikan permainan ini. Salah satunya adalah dengan algoritma backtracking, yang dibahas dalam makalah ini. Agar lebih mangkus, algoritma dibantu dengan teknik heuristic berdasarkan logika-logika yang digunakan jika dimainkan secara manual.**

***Kata Kunci* - game, nonogram, runut balik, NP-Complete**

I. PENDAHULUAN

Di dunia ini ada banyak permainan-permainan pengasah logika yang sudah diciptakan dan dikembangkan. Mulai dari permainan tabel sederhana seperti Sudoku dan Kakuro. Hingga permainan yang berbasis perangkat lunak seperti Go Figure, Tetravex, Klotski dan Solitaire. Sudah banyak orang yang membuat algoritma untuk membantu menyelesaikan permainan-permainan tersebut. Untuk itulah penulis tertariku untuk menyusun suatu algoritma penyelesaian untuk menyelesaikan permainan (sehingga solusi ini seakan berbasis kecerdasan buatan).

Permainan yang dipilih adalah nonogram, karena permainan ini cukup jarang diketahui sehingga masih sedikit orang yang membuat algoritma penyelesaiannya. Selain itu, permasalahan nonogram cukup kompleks karena sudah dibuktikan bahwa ini adalah masalah NP-Complete, yakni tidak bisa ditemukan algoritma dengan kompleksitas polynomial yang mampu menyelesaikan permasalahan di permainan ini.

Metode yang dirancang adalah dengan memanfaatkan algoritma backtracking, karena algoritma ini terbukti cukup ampuh dan mangkus serta sudah banyak digunakan. Selanjutnya, untuk mempercepat penemuan solusi, ditambahkan pula beberapa heuristik berdasarkan logika yang digunakan saat pemain biasa mencoba menyelesaikan permainan ini.

II. NONOGRAM

A. Definisi

Permainan nonogram diciptakan pada tahun 1987 oleh Tetsuya Nishio, dan dikenal dengan berbagai nama, antara lain: *griddler (grid riddle)*, *pixel puzzles*, *paint by numbers*, dsb.

Pada sebuah puzzle nonogram, terdapat sebuah matriks dua dimensi, dan setiap sel dapat diwarnai atau dibiarkan kosong berdasarkan sequence bilangan-bilangan sebagai "*clue*" yang tertera di sekeliling matriks tersebut. Bilangan-bilangan tersebut menyatakan kelompok sel berwarna yang harus berada pada sebuah baris atau kolom.

Contohnya, jika sebuah baris memiliki clue [3 1 2], maka pada baris tersebut harus terdapat sebuah kelompok yang terdiri dari 3 sel berwarna, diikuti oleh sebuah kelompok yang terdiri dari 1 sel berwarna, dan diakhiri dengan sebuah kelompok yang terdiri dari 2 sel berwarna. Masing-masing kelompok harus dipisah oleh sekurangnya sebuah sel yang kosong.

Biasanya, sebuah puzzle nonogram hanya memiliki sebuah solusi unik yang sesuai dengan semua clue yang diberikan. Jika semua sel telah ditentukan diisi warna atau kosong, maka solusi berupa sebuah gambar yang muncul pada matriks tersebut.

Jadi inti dari permainan ini adalah menentukan sel mana saja dalam matrik yang akan diwarnai berdasarkan petunjuk bilangan-bilangan yang diberikan di sekitar matrik. Sehingga kumpulan piksel tersebut akan membentuk gambar.

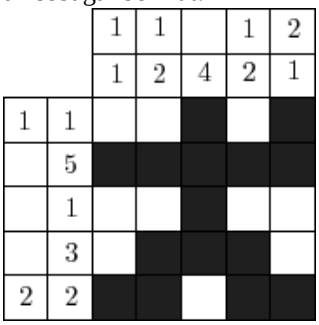
B. Contoh

		1	1		1	2
		1	2	4	2	1
1	1					
	5					
	1					
	3					
2	2					

Gambar 1. Contoh permasalahan nonogram

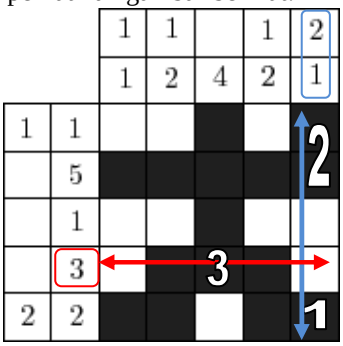
Gambar di atas adalah salah satu contoh sederhanna dari persoalan nonogram. Terlihat terdapat matrik utama berukuran 5x5. Sel-sel di matrik ini lah yang akan dihadapi pemain. Yakni tiap sel harus ditentukan, diwarnai atau tidak.

Petunjuk yang diberikan adalah angka-angka di sekitar matrik tersebut. Dari petunjuk tersebut, solusi yang diharapkan adalah sebagai berikut:



Gambar 2. Contoh solusi permasalahan nonogram

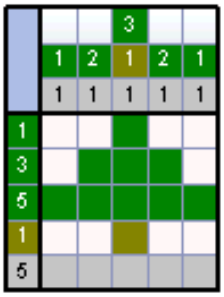
Terlihat bahwa angka-angka petunjuk dalam suatu baris/kolom menyatakan banyaknya sel di tiap-tiap kelompok yang dipisahkan oleh sel kosong. Lebih jelasnya, perhatikan gambar berikut:



Gambar 3. Arti dari angka-angka petunjuk

Sehingga, setelah diselesaikan akan terbentuk suatu gambar yang terdiri dari sel-sel yang diwarnai.

Dalam pengembangannya, pengisian warna di sel nonogram ini bisa bervariasi (tidak hanya satu warna), sehingga gambar yang dihasilkan bisa lebih menarik. Misalnya pada gambar berikut:



Gambar 4. Arti dari angka-angka petunjuk

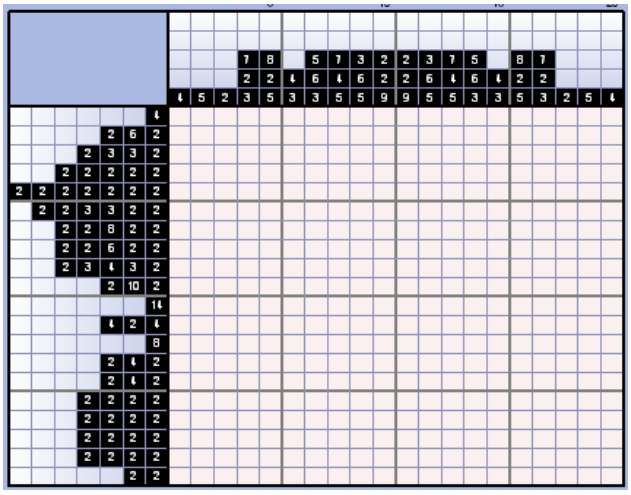
Namun, di makalah ini hanya dibahas nonogram yang terdiri dari satu warna, sehingga tiap sel bernilai biner (diwarnai atau tidak)

C. Penggunaan

Secara umum, memang permasalahan nonogram hanyalah sekedar untuk dijadikan permainan sebagai pengisi waktu luang sekaligus mengasah kemampuan otak. Permainan ini sering muncul sebagai teka-teki di koran harian, dan juga telah dijadikan videogame oleh Nintendo pada konsol Game Boy (Mario Picross) dan Nintendo DS (Picross DS).

Namun, jika dikembangkan lebih jauh, sebenarnya permasalahan ini juga bisa digunakan untuk kriptografi, yakni dengan menyembunyikan pesan di dalam gambar yang akan dihasilkan. Bisa berarti pesannya adalah gambar itu sendiri, atau bisa juga sel-sel di dalam matrik menyatakan bit-bit yang jika disusun atau dimanipulasi akan membentuk pesan yang bermakna.

Pada contoh-contoh di atas memang sepertinya permasalahan ini sederhana dan dapat diselesaikan dengan mudah. Namun semakin besar ukuran matriknya, usaha untuk menyelesaikannya akan jauh semakin berat juga. Misalnya pada contoh soal berikut:



Gambar 5. Contoh soal nonogram berukuran besar

Gambar tersebut adalah permasalahan nonogram dengan ukuran 20x20. Dan saat dicoba, akan terasa bahwa cukup sulit untuk memecahkan teka-teki tersebut.

III. ALGORITMA RUNUT BALIK

A. Definisi

Dalam penyelesaian masalah pencarian, dapat digunakan suatu pendekatan dengan graf. Di mana, masalah yang kita hadapi kita representasikan menjadi suatu pohon (graf). Dengan tiap simpul menyatakan state-state yang mungkin. Selanjutnya, kita lakukan traversal dari satu simpul ke simpul lain hingga menemukan state sesuai yang kita inginkan (solusinya). Terdapat dua jenis traversal pencarian yang umum digunakan, yakni pencarian melebar (BFS), dan pencarian mendalam (DFS).

Algoritma runut balik, alias *backtracking* adalah pengembangan dari algoritma DFS. Pada algoritma pencarian mendalam, ketika berada di suatu simpul, pada

setiap langkah kunjungi simpul lain yang bertetangga. Begitu seterusnya hingga menemukan solusi atau sudah sampai di daun (tidak bisa melangkah lebih jauh lagi). Jika mentok, melakukan **runut-balik**, yakni **kembali ke simpul sebelumnya**, lalu mengunjungi anak simpul yang lain. Lebih jelasnya:

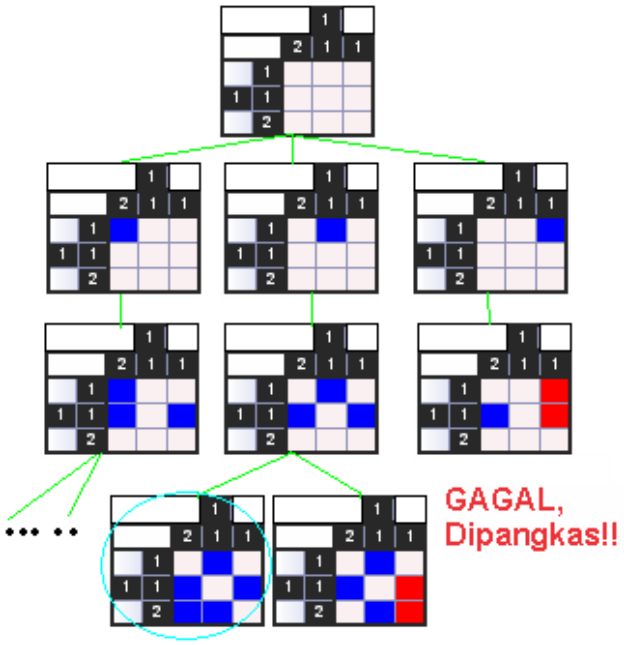
• Traversal dimulai dari simpul *v*.
• Algoritma:
1. Kunjungi simpul *v*,
2. Kunjungi simpul *w* yang bertetangga dengan simpul *v*.
3. Ulangi *DFS* mulai dari simpul *w*.
4. Jika tidak mengarah ke solusi, lakukan runut-balik

B. Aplikasi Backtracking untuk Nonogram

Pada permainan, konversi menjadi pohon ruang status adalah sebagai berikut:

- ✓ Akar dari pohon adalah matrik yang masih kosong
- ✓ Simpul menyatakan kondisi pewarnaan matriks sementara yang sudah dilakukan
- ✓ Kedalaman menyatakan baris di matrik yang sedang ditangani. Sehingga batas kedalaman adalah lebar matrik.
- ✓ Simpul-simpul pada satu aras menyatakan kemungkinan-kemungkinan susunan pewarnaan pada suatu baris. Pada suatu baris, dipastikan pewarnaan sesuai dengan angka petunjuk dibaris tersebut, sehingga pemeriksaan valid atau tidaknya tinggal dibandingkan dengan angka-angka petunjuk di tiap kolom.

Untuk lebih jelasnya, perhatikan contoh pohon ruang status berikut:



Gambar 6. Contoh pohon ruang status permainan nonogram

Ketika pada suatu simpul, terjadi pewarnaan yang tidak valid (pada gambar di atas ditandai dengan warna merah), maka akan dilakukan *backtracking*, yakni kembali ke aras sebelumnya. Begitu juga ketika sudah mencapa kedalaman paling bawah, tetapi solusi belum ditemukan.

IV. METODE LOGIKA

Dalam permainan nonogram ini, dapat ditemukan beberapa metode logika untuk mempercepat penemuan solusi, yakni dengan cara langsung menandai terlebih dahulu sel-sel yang pasti akan diwarnai. Berikut trik-trik untuk menentukan sel yang pasti diwarnai hanya dengan memperhatikan angka petunjuk di suatu baris/kolom. Trik ini biasa digunakan para pemain untuk mengawali pencarian solusinya. (Perhatian bahwa trik-trik ini adalah hasil pemikiran penulis sendiri, sehingga bukan merupakan aturan yang formal)

Disini C1, C2, … Ck angka-angka petunjuk di suatu baris (ada sebanyak k angka), dan N menyatakan banyak sel dalam satu baris.

Kasus 1: Single Complete Fill

Yaitu saat k=1, dan C1=N, yakni angka clue sama dengan banyak sel sebaris, sehingga jelas bahwa seluruh sel dalam satu baris akan diwarnai

Gambar 7. Contoh kasus single complete fill dengan C1=N=5

Kasus 2: Single Partial Fill

Dapat diterapkan saat k=1 yang memenuhi C1 > N/2
Sehingga menurut pigeon hole principle, akan ada minimal satu sel yang jelas diwarnai.
Untuk lebih jelasnya perhatikan contoh untuk C1=4 dan N=6 berikut. Maka ada 3 kemungkinan pewarnaan:

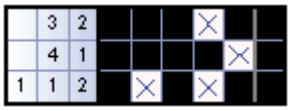
Gambar 8. tiga kemungkinan pewarnaan untuk C1=4 dan N=6

Di antara ketiga kemungkinan tersebut, terlihat bahwa sel ke 3 dan 4 pasti diwarnai.

Gambar 9. Untuk C1=4 dan N=6, ada 2 sel yang pasti diwarnai

Kasus 3: Complete Fills

Yaitu saat ΣCi + k – 1 =N, dengan kata lain, jika semua angka petunjuk dijumlahkan, lalu ditambah dengan celah kosong di antaranya, sudah cukup untuk memenuhi sel-sel sebaris.



Gambar 10. Contoh-contoh kasus complete fills

Kasus 4: Partial Fills

Mirip dengan kasus 2, hanya saja disesuaikan dengan celah karena banyak angka lebih dari satu. Perhatikan contoh berikut:

Gambar 11. contoh semua kemungkinan di kasus partial fills

Di antara semua kemungkinan tersebut, terlihat bahwa sel ke 4 dan 5 pasti diwarnai.

Gambar 12. Sel-sel yang pasti di warnai

Proses penerapan trik-trik ini biasa dilakukan di awal, sehingga cukup memeriksa angka-angka petunjuk di tiap baris dan kolom masing-masing sekali. Setelah menerapkan teknik tersebut, akan didapatkan sel-sel yang pasti diwarnai.

V. IMPLEMENTASI

Berikut pseudocode dari nonogram solver yang dibuat:

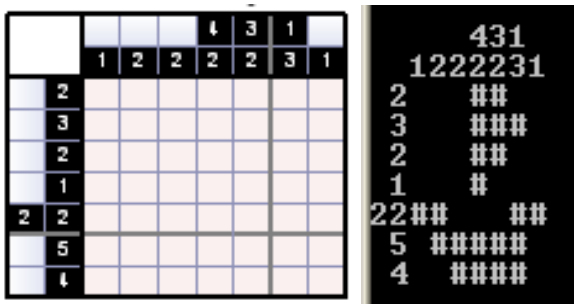
```
Program nonogram_solver()
{
Program untuk menemukan solusi dari suatu permainan nonogram dengan algoritma runut balik
}

Deklarasi

Procedure set_posisi_awal(integer b)
//inisiasi pewarnaan di baris b, yakni semua grup dibuat dan masing-masing dipisahkan dengan sebuah sel kosong

Function isAvailable(b: integer) → Boolean
//memeriksa apakah pewarnaan di b baris teratas sudah valid

type clue = record
(angka: integer, awal: integer)

H : integer //lebar matrik
W : integer //panjang matrik

diwarnai : array [1..H][1..W] of boolean
//status warna tiap-tiap isi matrik

k_bar : array [1..H] of integer
k_kol : array [1..W] of integer
//banyak clue di suatu baris atau kolom

Cbar : array [1..H][1..k_bar] of clue
Ckol : array [1..W][1..k_kol] of clue
//clue di tiap-tiap baris atau kolom

solvable: Boolean
//false jika tidak ada solusi

solved : Boolean
//true jika solusi sudah ditemukan

baris : integer //current baris
c_g : integer //current group

ALGORITMA

Solvable ← true
solved ← false

baris ← 1
c_g ← k_bar[baris]
set_posisi_awal(baris)

repeat
 if(!isAvailable(baris))
  if(c_g adalah grup terkhir)
   if(grup terakhir belum mencapai ujung)
    geser grup terakhir ke kanan
   else if(c_g>1)
    c_g ← c_g -1
   else
    solvable ← false
   endif
  else if (grup sebelumnya tidak bisa maju)
   if(c_g>1)
    c_g ← c_g -1
   else if(baris = 1)
    solvable ← false
   else
    LAKUKAN BACKTRACKING
   endif
  else
   geser grup ini ke kanan
   rapatkan semua grup di kanan grup ini
  endif
 else if (baris<H)
  baris ← baris + 1
  set_posisi_awal(baris)
 else solved← true
 endif
until (!solvable OR solved)
```

Untuk meningkatkan performansi, pada proses penggeseran grup, sebenarnya tidak perlu menggeser satu per satu pewarnaan sel, namun cukup meenghapus warna di satu ujung , lalu mewarnai sel di ujung lainnya. Berikut contoh kodenya:

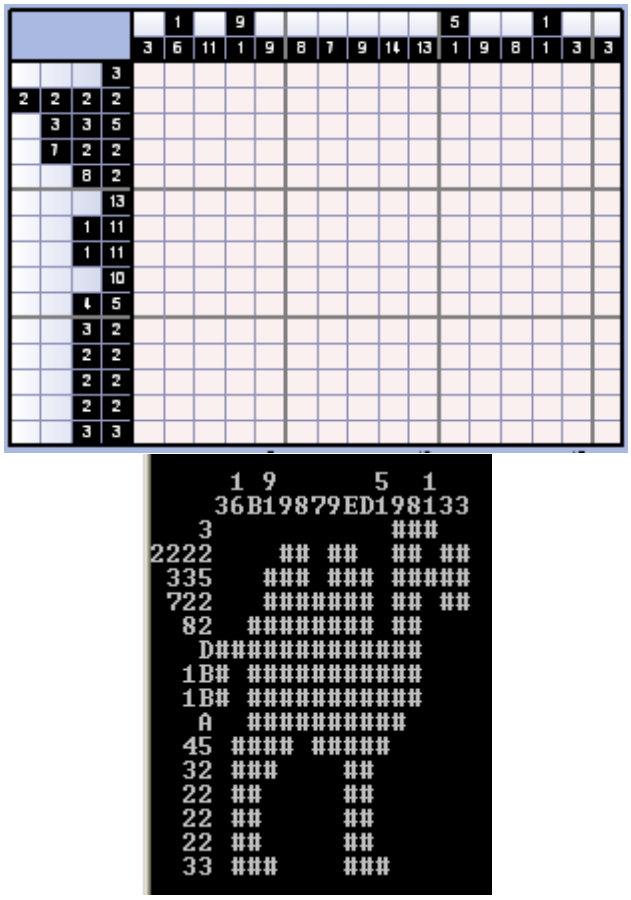
```
diwarnai[baris][Cbar[baris][c_g].awal] ← false
diwarnai[baris]
[Cbar[baris][c_g].awal+Cbar[baris][c_g].angka] ← true
Cbar[baris][c_g].awal = Cbar[baris][c_g].awal + 1
//menggeser suatu grup ke kanan
```

VI. PENGUJIAN DAN ANALISIS

Dari program yang telah penulis rancang, berikut hasil pengujian dari beberapa soal nonogram:



Gambar 13. contoh soal nonogram dan solusi oleh program



Gambar 14. contoh soal nonogram dan solusi oleh program

Permasalahan nonogram ini cukup kompleks. Kasus terburuknya, jika diselesaikan dengan algoritma brute force, berhubung tiap sel ada dua kemungkinan (diwarnai atau tidak), dan ada sebanyak MN sel (untuk ukuran matrik MxN), maka dapat disimpulkan bahwa kompleksitasnya adalah **O(2^{MN})**. Memang sangat tidak mangkus, masih eksponensial.

Sedangkan dengan algoritma runut balik, solusinya bisa lebih baik. Di tiap baris maksimal ada N kombinasi, sedangkan jumlah barisnya ada M. Sehingga dengan algoritma ini, kompleksitasnya menjadi **O(M^N)**. Memang masih eksponensial, tapi setidaknya memperbaiki solusi brute force. (ingat bahwa $2^{MN} = (2^M)^N > M^N$ yang ekivalen dengan $2^M > M$, yang selalu benar untuk $M \geq 1$)

Nonogram sudah dibuktikan merupakan masalah **NP-complete**. Oleh karena itu, kompleksitas eksponensial sudah tergolong cukup akurat untuk menyelesaikan permainan ini. Dan masih ada kemungkinan untuk menemukan algoritma yang lebih mangkus lagi, bahkan yang polynomial.

VII. KESIMPULAN

Dari pembahasan di bab-bab sebelumnya, terlihat bahwa permainan nonogram/griddles cukup menarik dan dapat mengasah daya logika. Karena memang permainan ini tergolong masalah NP-complete sehingga harus dibuat suatu algoritma semangkus mungkin untuk menyelesaikannya.

Penyelesaian biasa dengan algoritma brute force cukup boros karena kompleksitasnya O(2^{MN}). Salah satu alternatif yaitu menggunakan algoritma runut balik (*backtracking*) yang memberikan kompleksitas **O(M^N).** Namun, sepertinya belum ditemukan algoritma polinomial untuk penyelesaian permasalahan ini.

REFERENSI

[1] Munir, Rinaldi, "Diktat Kuliah IF2251 Strategi Algoritmik", Program Studi Teknik Informatika, ITB, 2007.

[2] http://en.wikipedia.org/wiki/Nonogram diakses tanggal 6 Desember 2010, 17.00

[3] Game Griddlers Deluxe 2007

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, 7 Desember 2010

Hendra Hadhil Choiri
135 08 041