Penerapan Algoritma *Breadth First Search* dalam Pencarian Solusi *Knight Game*

Novi Setiani[1], Mastura Diana Marieska[2], Thazin Aungsoe[3]

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail : if114119@students.if.itb.ac.id[1], if14136@students.if.itb.ac.id[2], if14163@students.if.itb.ac.id[3]

Abstrak

Game teka-teki adalah salah satu jenis *game* yang menantang pemain untuk mencari solusi dari suatu persoalan. Contohnya adalah *Knight game*. Dalam *knight game*, pemain yang direpresentasikan sebagai seorang kesatria berkuda ditantang untuk melewati sejumlah kotak dan kembali lagi pada kotak awal. Tiap kotak hanya boleh dilewati satu kali dan gerakan yang dapat dilakukan adalah gerakan kuda catur. Pemain bebas memilih kotak mana yang akan dijadikan kotak awal. Untuk menemukan solusi, kami mencoba menggunakan algoritma BFS (*Breadth First Search*). Algoritma BFS yang mengeksplorasi tiap cabang dari tiap *node* dinilai mangkus dalam mencari solusi *knight game*. Kami menemukan bahwa solusi persoalan ini tidak bergantung pada kotak awal. Solusi persoalan bergantung pada kalang gerakan yang harus ditempuh pemain. Jika suatu kalang telah ditemukan, maka pemain dapat memulai dari kotak manapun. Selama pemain mengikuti kalang, pemain akan selalu kembali pada kotak awal.

***Kata kunci**: Knight game, gerak kuda catur, BFS, kalang*

1. Pendahuluan

Teknologi *game* kini berkembang pesat sejalan dengan perkembangan teknologi informasi. Tidak semua *game* hanya mengandalkan ketangkasan pemain dalam menggerakkan tetikus atau menekan tombol-tombol *keyboard*. Ada *game* yang mengandalkan kemampuan logika pemain untuk menemukan solusi suatu persoalan.
Knight game menantang pemain untuk menyeberang ke suatu kastil melewati jembatan yang terdiri atas 14 kotak. Pemain bebas memilih kotak pertama untuk memulai permainan, namun harus dapat kembali lagi ke kotak tersebut. Tiap kotak harus dan hanya boleh dilewati satu kali. Gerakan yang boileh dilakukan adalah gerakan kuda catur yang membentuk huruf L seperti ditunjukkan pada gambar.

Keterangan : Kotak berwarna ungu adalah kotak yang dapat dicapai dari posisi kesatria

2. Terminologi

1. Status persoalan (*problem state*)
   Simpul-simpul di dalam pohon dinamis yang memenuhi kendala (*constraints*).
2. Status solusi (*solusi state*)
   Semua simpul telah dikunjungi.
3. Ruang solusi (*solution space*)
   Himpunan solusi.

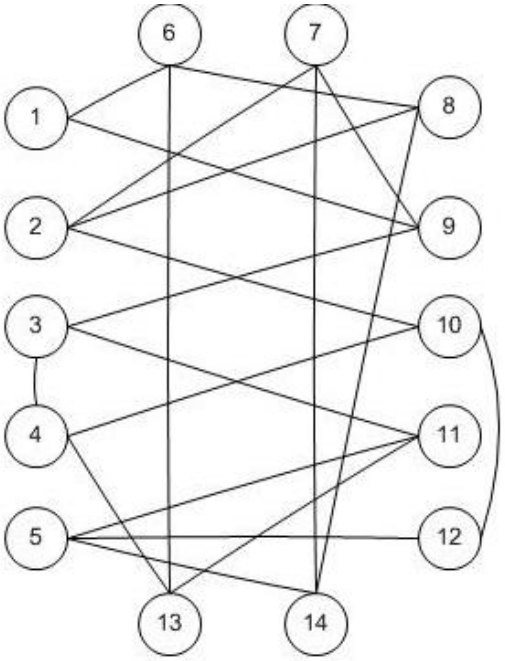
Peraturan *Knight game* adalah sebagai berikut :
Seorang kesatria ingin menyeberang papan dengan 14 kotak. Kesatria tersebut harus mengunjungi satu kotak tepat satu kali dan kembali ke kotak awal. Kesatria akan berjalan seperti langkah kuda di papan catur. Kami mencoba menyelesaikan persoalan ini dengan menggunakan BFS (*Breadth First Search*). Dalam implementasinya, diperlukan struktur data stack dan pohon.

3. Graf Ruang Solusi

Jika kotak-kotak yang harus diseberangi diberi penomoran seperti berikut ini

1	2	3	
4	5	6	7
8	9	10	11
12	13	14	

Maka akan dihasilkan graf ruang solusi sebagai berikut



Graf tidak berarah ini yang kami jadikan dasar permasalahan untuk dicari solusinya menggunakan algoritma BFS.

4. *Pseudo Code* dan prosedur

Penerapan BFS dalam permasalahan ini kami nyatakan dalam satu prosedur `jalanKnight`

```
Procedure jalanKnight(int nodeAwal)

        Push nodeAwal ke stack
        N=0
        While (n<14) and (anakcurrentNode=/
        node1)
                Pop kepala stack
                Bangkitkan semua simpul tetangga
                For (semua anak yg dibangkitkan)
                        Isparent(node,cekp,n)
                        If  ( not isparent) then
                                Push ke stack
                        Else
                                Bunuh node
                        Endif
                Endfor
        Endwhile

        if (Tetangga(CurrentNode) == NodeAwal)
        then Solusi ditemukan
        else
        {jalur yang ditempuh tidak dapat kembali
        ke nodeAwal}

End procedure

Procedure isParent(int node,boolean cekp,int n)

        Temp= node
        cekp =false
        N=0
        While (temp =/ null) and (not cekp)
                If (parent (temp) =/ node) then
                        Temp =parent(node)
                        N++
                Else
                        Cekp=true
                Endif
endwhile
```

Penjelasan algoritmanya sebagai berikut:

Fungsi utama :

Prosedur `JalanKnight(int nodeAwal)`

Prekondisi : .Dalam prosedur ini, kami mengambil asumsi bahwa telah ada struktur data berupa matriks ketetanggaan untuk merepresentasikan ketetanggaan antar simpul.

1. Memasukkan salah satu *node* ke dalam antrian S dan set kedalaman pohon dengan 0.
2. Mengambil kepala antrian S
3. Bangkitkan semua simpul anak *node*
4. Masukkan *parent* ke semua anak *node*
5. Jika ada anak yang belum pernah dibangkitkan ,masukkan ke dalam stack.
6. Jika sudah pernah dibangkitkan dalam jalur yang sama, didealokasi.
7. Jika kedalaman belum sampai 14 ,ulangi langkah 2.

Fungsi bantuan:

1. BangkitAnak
   Prosedur pembangkitan anaknya mengikuti aturan permainan knight yaitu seperti langkah kuda pada permainan catur.

2. Prosedur isParent(int node, boolean cekp, int n)
   Sebelum anak dari current *node* dimasukkan ke dalam stack , semua anak *node* harus dicek apakah pernah dibangkitkan di jalur yang sama atau belum. Untuk mengeceknya, kita menggunakan procedure `isParent()` yang parameter keluarannya adalah tipe *boolean* dan *depth* (kedalaman pohon).

Contoh:

	6		1
7	2		8
9		3	5
		4	

Node 1 dimasukkan ke dalam antrian S. Karena kedalaman pohonnya belum 14, maka bangkitkan anak-anaknya. Dalam kasus ini, anak-anak yang bisa dibangkitkan adalah 2 dan 3 sesuai posisi di bawah. Lalu, *node* 2 dan 3 dicek apakah sudah pernah dibangkitkan dalam jalur yang sama.

			1
	2		
		3	

Sesuai metode BFS, maka langkah selanjutnya adalah mengecek *node*2 dan membangkitkan anak-anaknya yaitu 4 dan 5.

			1
	2		
		3	5
		4	

Kemudian *node* 3 dicek dan anak-anaknya yakni 6 dan 7 dibangkitkan.

	6		1
7	2		
		3	5
		4	

Dari *node* 4 dicek kedalamannya, jika belum 14 maka dibangkitkan anak-anaknya yakni 8 dan 9

Begitu seterusnya hingga didapatkan solusi dengan urutan langkah dari *node* 1 sebagai berikut:

8	5	12	1
13	2	9	6
4	7	14	11
	10	3	

Dari langkah ke 14 dapat kembali lagi ke simpul awal yakni simpul 1.

5. Kesimpulan

Algoritma BFS kami nilai tidak cukup mangkus untuk menemukan solusi dari Knight *game* ini. Setelah kami mencoba mencari solusi secara manual dengan metode *Depth First Search*, ditemukan lebih dari satu kalang solusi. Untuk persoalan yang memiliki lebih dari satu solusi, algoritma DFS lebih mangkus daripada algoritma BFS karena DFS menemukan solusi setelah mengeksplorasi hanya sebagian kecil dari seluruh ruang status. Sedangkan BFS masih harus menelusuri semua lintasan pada aras n -1 sebelum memeriksa solusi pada aras n.

Penyelesaian *game* ini tidak bergantung pada kotak yang dipilih sebagai kotak awal, namun bergantung pada kalang solusi. Dari kotak manapun pemain memulai *game*, pemain tetap dapat kembali ke kotak awal jika mengikuti alur dari kalang solusi.

Daftar Pustaka

1. Munir, Rinaldi. "*Strategi Algoritmik*", Lab Ilmu Rekayasa Komputasi, Departemen Teknik Informatika ITB.2005

2. **http://www.plastelina.net**
   Diakses tanggal 13 Mei 2006 pukul 10.45