Penyelesaian Permasalahan *Knight's Tour* Menggunakan Algoritma *Breadth First Search* (BFS)

Fahmi Mumtaz and 13506045[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*if16045@itb.ac.id*

Abstract*— Makalah ini membahas tentang algoritma untuk menyelesaikan permasalahan pada aplikasi teori graf di permainan catur papan. Permasalahan yang dibahas pada makalah ini adalah masalah knight's tour. Ada banyak algoritma yang dapat dipakai untuk menyelesaikan permasalahan tersebut. Tapi dalam makalah ini, algoritma yang dipilih oleh penulis adalah algoritma Pencarian Melebar (Breadth First Search).

***Index Terms*—Knight's tour, breadth first search, pohon, algoritma.**

I. PENDAHULUAN

Di dalam kehidupannya, manusia selalu menemui masalah atau persoalan. Hal ini mungkin didasarkan dari sifat dasar manusia itu sendiri yang selalu ingin tahu tentang segala sesuatu. Deskripsi dari masalah menurut [1] adalah pertanyaan atau tugas yang kita cari jawabannya.

Dalam menghadapi permasalahan, untuk menyelesaikannya manusia memerlukan langkah-langkah yang benar sehingga permasalahan tersebut dapat terselesaikan. Urutan langkah-langkah untuk memecahkan suatu masalah tersebutlah yang dinamakan dengan algoritma. Definisi lain dari algoritma adalah deretan langkah-langkah komputasi yang mentransformasikan data masukan menjadi keluaran [1].

Dalam menentukan langkah-langkah tersebut diperlukan suatu strategi agar langkah-langakah yang dipakai tersebut dapat menyelesaikan permasalahan secara mangkus (efisien). Strategi menurut Kamus Besar Bahasa Indonesia (KBBI) edisi tahun 1998 adalah rencana yang cermat mengenai kegiatan untuk mencapai sasaran khusus. Rencana itu sendiri dapat berisi suatu metode atau teknik yang digunakan untuk mencapai sasaran khusus tersebut.

Dengan pengertian algoritma dan strategi tersebut, kita dapat mendefinisikan strategi algoritmik (algorithm strategies) sebagai kumpulan metode atau teknik untuk memecahkan masalah guna mencapai tujuan yang ditentukan, yang dalam hal ini deskripsi metode atau teknik tersebut dinyatakan dalam suatu urutan langkah-langkah penyelesaian.

Secara umum, strategi pemecahan masalah dapat dikelompokan menjadi:

1. Strategi solusi langsung (*direct solution strategies*)
   Metode yang termasuk ke dalam strategi ini adalah:
   - Algoritma *Brute Force*
   - Algoritma *Greedy*
2. Strategi berbasis pencarian pada ruang status (*state-space bace strategies*)
   Metode yang termasuk ke dalam strategi ini adalah:
   - Algoritma *Backtracking*
   - Algoritma *Branch and Bound*
3. Strategi solusi atas-bawah (*top-down solution strategies*)
   Metode yang termasuk ke dalam strategi ini adalah algoritma *Divide and Conquer*.
4. Strategi traversal atau mengunjungi simpul-simpul secara sistematik.
   Metode yang termasuk ke dalam strategi ini adalah:
   - Algoritma *Breadth First Search*
   - Algoritma *Depth First Search*
5. Strategi solusi bawah-atas (*bottom-up solution strategies*)
   Metode yang termasuk ke dalam strategi ini adalah *Dynamic Programming.*

Strategi yang akan dipakai dalam pemecahan masalah dalam makalah ini adalah strategi berbasis pencarian traversal. Secara spesifik, metode yang digunakan adalah algoritma pencarian melebar (*Breadth First Search*).

Permasalahan yang dibahas disini adalah permasalahan dari aplikasi teori graf pada permainan catur papan yang bernama Knight's Tour. Permasalahan ini akan dibahas lebih lanjut dalam bab selanjutnya.

II. *KNIGHT'S TOUR*

Knight's Tour adalah suatu aplikasi dari teori graf pada permainan catur papan. Suatu Knight's Tour pada papan catur adalah rangkaian perjalanan kuda catur pada papan catur sehingga seluruh kotak terlewati kuda tepat satu kali [2]. Aturan langkah kuda pada permainan catur adalah sebagai berikut :

- Melangkah dua persegi ke arah horisontal kemudian satu persegi ke arah vertikal, atau

- Melangkah dua persegi ke arah vertikal kemudian satu persegi ke arah horisontal, atau
- Melangkah dua persegi ke arah vertikal kemudian satu persegi ke arah horisontal, atau
- Melangkah satu persegi ke arah vertikal kemudian dua persegi ke arah horisontal.

Jika dalam *Knight's Tour* setiap persegi dari papan catur dapat dilewati tepat satu kali dan kuda kembali pada persegi semula maka disebut langkah kuda tertutup (*Closed Knight's Tour*). Namun, jika semua persegi telah dilewati dan kuda tidak dapat kembali ke posisi semula maka disebut langkah kuda yang terbuka (*Open Knight's Tour*)

4	1	6	55	10	51	46	49
7	56	3	64	45	48	11	52
2	5	58	9	54	13	50	47
57	8	63	14	59	44	53	12
28	37	60	43	62	15	24	41
31	34	29	38	25	42	21	18
36	27	32	61	16	19	40	23
33	30	35	26	39	22	17	20

Gambar 1 Contoh *Closed Knight's Tour*

pada papan catur ukuran 8 x 8

Gambar 1 di atas adalah contoh dari *Knight's Tour*. Angka 1-64 menandakan urutan kotak yang dilewati oleh kuda catur tersebut. Langkah yang diambil oleh kuda urut dari 1 sampai dengan 64.

Banyak ilmuwan yang tertarik dengan permasalahan ini diantaranya adalah Leonhard Euler, Paul de Hijo, Sainte-Marie, Louis Posa, Allen Schwenk, Mark R. Keen, M. Kraitchik, Sloane's, Wegener, dan masih banyak ilmuan lainnya.

III. GRAF

Berdasarkan [3], konsep graf Eulerian yang diawali oleh karya Euler pada problem Jembatan Konigsberg (1735) merupakan awal dari lahirnya teori graf. Meskipun umurnya yang relatif muda, teori graf sebagai cabang dari matematika diskrit telah berkembang sangat pesat akhir-akhir ini, baik dalam pengembangan teori maupun aplikasi di berbagai bidang. Tidak hanya ilmuan dari luar negeri yang tertarik dalam mempelajarinya. Ilmuan dari Indonesia juga memiliki kontribusi dalam perkembangan teori tersebut. Adalah seorang Guru Besar ITB, Prof. Edy Tri Baskoro, yang memiliki kontribusi dalam pengembangan teori graf ini. Di sadari atau tidak, banyak aplikasi teori graf dalam kehidupan kita. Banyak sekali struktur yang bisa direpresentasikan dengan graf dan banyak masalah yang bisa diselesaikan dengan bantuan graf, bahkan dalam permainan catur pun ternyata ada aplikasi teori graf.

Secara informal, suatu graf adalah himpunan benda-benda yang disebut verteks (atau *node*) yang terhubung oleh sisi (atau *edge* atau *arc*). Biasanya graf digambarkan sebagai kumpulan titik-titik (melambangkan verteks) yang dihubungkan oleh garis-garis (melambangkan sisi).

Secara formal atau dalam bahasa matematika :

Graf $G = (V,E)$, yang dalam hal ini :

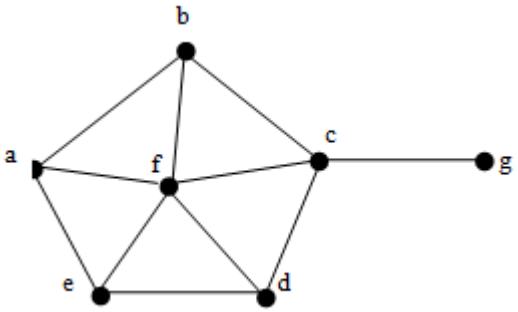
V = himpunan tidak-kosong dan berhingga dari simpul-simpul (*vertices*)
= $\{v_1, v_2, \ldots, v_n\}$

E = himpunan sisi (*edges*) yang menghubungkan sepasang simpul
= $\{e_1, e_2, \ldots, e_n\}$

Order dari graf G, ditulis dengan notasi |V(G)|, menyatakan banyakya titik (simpul) pada graf *G*.

Pada graf *G*, *jalan J* dari titik v_0 ke titik v_n adalah suatu barisan selang-seling dari titik dan sisi $v_0, e_0, v_0, \ldots, v_{n-1}, e_{n-1}, v_n$ yang dimulai dan diakhiri dengan titik, dengan sisi $e_1 = v_iv_{i+1}$ untuk $i = 0,1,2, \ldots, n$ sedemikian sehingga $v_iv_{i+1} \in E(G)$. *Panjang* dari jalan $v_0, e_0, v_0, \ldots, v_{n-1}, e_{n-1}, v_n$ adalah banyaknya sisi pada barisan tersebut. Titik v_0 dan v_n disebut titik-titik ujung dari jalan tersebut. Jika pada jalan *J* berlaku $v_0 = v_n$ maka *J* disebut *jalan tertutup* dan dikatakan *jalan terbuka* jika $v_0 \neq v_n$.

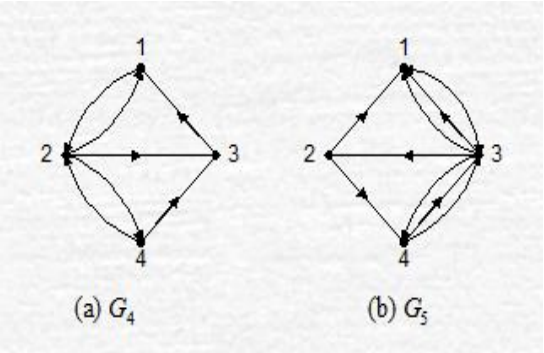
Jalan *J* disebut *lintasan* (*path*) bila semua titiknya berbeda. Sedangkan jika setiap sisinya yang berbeda maka jalan tersebut dinamakan *jejak* (*trail*). Jejak tertutup disebut *sirkit*. Sirkuit yang semua titiknya berlainan disebut siklus (*cycle*).



Gambar 2 Contoh Graf

Pada gambar 2 di atas, afbfcdc merupaka jalan (*walk*), abfcgcfea merupakan jalan tertutup (*close walk*), afcdfb merupakan jejak (*trail*), aefcg merupakan lintasan (*path*), afedcfba merupakan sirkit, dan afbcdea merupakan siklus (*cycle*).

Berdasarkan orientasi arah pada sisi, graf dapat dibagi menjadi graf berarah dan graf tak berarah. Graf tak berarah adalah graf yang sisinya tidak mempunyai orientasi arah. Sebaliknya, graf berarah adalah graf yang sisinya memiliki arah. Gambar 2 di atas adalah contoh graf tak berarah. Sedangkan gambar 3 di bawah adalah contoh graf berarah.



Gambar 3 Graf (a) dan (b) adalah graf berarah

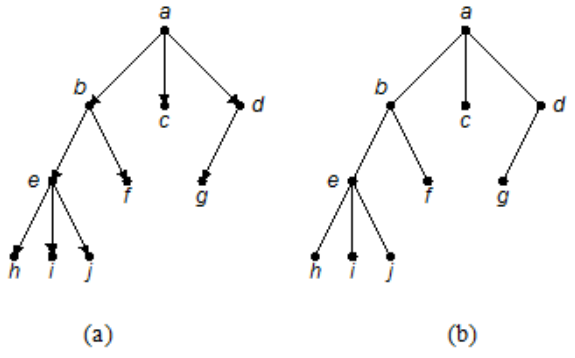
IV. POHON

Pohon (*tree*) adalah graf tak berarah terhubung yang tidak mengandung sirkuit [4]. Secara formal, pohon didefinisikan sebagai berikut:
Misalkan G = (V , E) adalah graf tak berarah sederhana dan jumlah simpulnya *n*, maka semua pernyataan di bawah ini adalah ekivalen:

- G adalah pohon.
- Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
- G terhubung dan memiliki $m = n - 1$ buah sisi.
- G tidak mengandung sirkuit dn memiliki $m = n - 1$ buah sisi.
- G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
- G terhubung dan semua sisinya adalah jembatan.

A.Pohon Berakar

Pohon yang satu buah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah disebut pohon berakar (*rooted tree*). Contoh pohon berakar dapat dilihat pada gambar 4 dibawah ini:



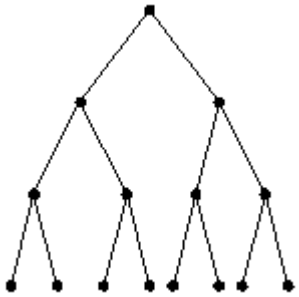
Gambar 4 (a) pohon berakar (b) sebagai perjanjian, tanda panah pada sisi dapat dibuang

Terdapat beberapa terminologi pada pohon berakar. Terminologi tersebut adalah:

- Orang tua (*parent*) dan anak (*child* atau *children*). Pada gambar 4 di atas *b*, *c*, *d* adalah anak-anak dari *a*. Sedangkan *a* adalah orang tua dari *b*, *c*, dan *d*.
- Lintasan (*path*). Pada gambar 4, lintasan dari *a* ke *j* adalah *a*, *b*, *e*, *j* dan panjang lintasannya adalah 4.
- Saudara kandung (*sibling*). *f* adalah saudara kandung dari *e*, tetapi *g* bukan saudara kandung dari *e* karena orang tua mereka berbeda.
- Derajat (*degree*). Derajat dari sebuah simpul adalah jumlah anak pada simpul tersebut.
- Daun (*leaf*). Daun adalah simpul yang berderajat nol atau tidak memiliki anak.

B. Pohon n-Ary

Pohon *n*-Ary adalah pohon berakar yang setiap simpul cabangnya mempunyai paling banyak *n* buah anak. Pohon *n*-ary dikatakn penuh atau teratur jika setiap simpul cabangnya mempunya tepat *n* buah anak. Gambar 5 adalah contoh gambar pohon n-Ary dengan n sejumlah 2 atau biasa disebut dengan pohon 2-Ary atau pohon biner.

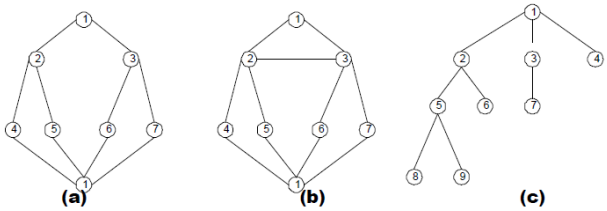


Gambar 5 Gambar pohon 2-Ary atau pohon biner

V. ALGORITMA BREADTH FIRST SEARCH

Berdasarkan [3], Algoritma *Breadth First Search* (BFS) atau sering juga disebut sebagai algoritma pencarian melebar adalah algoritma pencarian graf yang dimulai dari suatu simpul dan mengunjungi semua simpul yang bertetangga. Semua simpul-simpul yang bertetangga tersebut, dikembangkan semua simpul-simpul yang bertetangga dengan simpul tersebut, begitu seterusnya. Jika kita mengkhususkan pada graf berbentuk pohon, algoritma pencarian melebar ini adalah algoritma pencarian yang dimulai dari simpul akar dan mengunjungi semua simpul yang menjadi anak dari akar tersebut. Lalu dari semua simpul anak tersebut, dikembangkan untuk dikunjungi simpul-simpul anak dari simpul-anak-dari-akar tersebut, begitu seterusnya.

BFS adalah metode pencarian yang bertipe tidak memilik informasi (*uninformed*) yang bertujuan untuk mengembangkan dan memeriksa semua simpul dari graf atau kombinasi urutan dengan cara mencari secara sistematis melalui semua solusi. Dengan kata lain, metode ini mencari secara mendalam seluruh graf atau urutan tanpa mempertimbangkan tujuan sampai menemukannya.



Gambar 6 Contoh algoritma BFS pada tiga graf (a), (b), dan (c) beserta urutan simpul yang dilewati

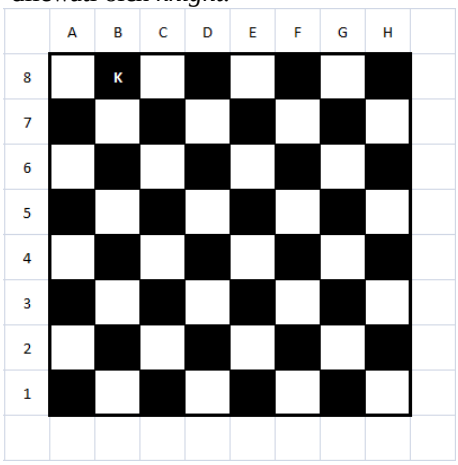
Algoritma BFS pada sebuah graf adalah sebagai berikut:

- Kunjungi simpul *v* (transversal dimulai dari simpul ini).
- Kunjungi semua simpul yang bertetangga dengan simpul *v* terlebih dahulu.
- Kunjungi simpul-simpul yang belum dikunjungi dan bertetangga dengan simpul-simpul yang tadi telah dikunjungi.
- Demikian seterusnya.

IV. PEMODELAN MASALAH KNIGHT'S TOUR DALAM REPRESENTASI POHON

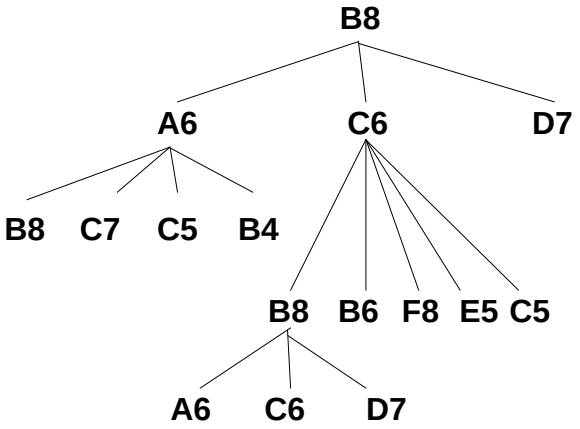
Permasalahan *knight's tour* dapat dipetakan dalam representasi pohon. Secara formal, representasi pohon untuk permasalahan *knight's tour* dapat dilihat sebagai berikut:

- **Orangtua (*parent*)** adalah posisi awal *knight* pada papan catur.
- **Anak (*child*)** adalah opsi kotak yang dapat ditempati *knight* untuk langkah selanjutnya.
- **Lintasan (*path*)** adalah daftar kotak yang telah dilewati oleh *knight*.



Gambar 7 Papan catur 8 x 8

Pada gambar 7 di atas dapat dilihat bahwa kotak B8 adalah orangtua pada permasalahan knight tour di atas dalam representasi pohon. Yang bertindak sebagai anak yang akan dikembangkan adalah kotak-kotak yang dapat dikunjungi oleh knight selanjutnya yaitu kotak A6, C6, dan D7. Begitu seterusnya. Dalam representasi pohon dapat digambarkan pada gambar 8 di bawah ini.



Gambar 8 Representasi dalam pohon sederhana

V. APLIKASI ALGORITMA BFS PADA KNIGHT'S TOUR

Algoritma BFS dapat dipakai untuk menyelesaikan permasalahan dari *knight's tour*. Ide dasar dari algoritma ini adalah dengan membangun solusi-solusi parsial dari masalah (dalam hal ini jalan selangkah pada papan) lalu mencoba memperluas solusi tersebut. Jika solusi yang telah dikembangkan membawa *knight* pada kotak yang telah dikunjungi maka simpul (kotak) tersebut tidak akan dikembangkan lagi dan akan memperluas solusi parsial lainnya.

Algoritma BFS yang dipakai adalah:

1. Dari kotak tempat kuda tersebut berada, dibangkitkan langkah-langkah berikutnya yang memungkinkan dilalui oleh kuda tersebut.
2. Salah satu langkah (kotak) dipilih untuk diperluas.
3. Kuda melangkah ke kotak yang dipilih tersebut dan mengecek apakah kotak tersebut telah dikunjungi.
4. Kembali ke langkah dua sampai langkah yang diperluas tidak dapat mencapai solusi (kuda tidak dapat melangkah lagi).
5. Apabila semua pilihan telah dikunjungi, pilih satu kotak (yang dibangkitkan pada langkah 1) yang paling pertama dikunjungi dan bangkitkan langkah-langkah berikutnya yang memungkinkan dilalui oleh kuda tersebut.
6. Lakukan kembali langkah 2 – 4.
7. Lakukan langkah 5 untuk semua opsi kotak pada langkah 1.
8. Pencarian langkah dihentikan bila telah melakukan solusi atau tidak ada langkah yang memungkinkan lagi bagi kuda catur tersebut.

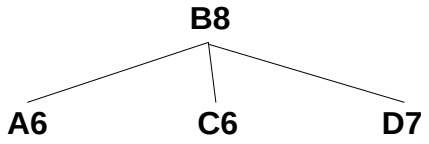
Contoh penggunaan algoritma BFS ini adalah sebagai berikut:

Dengan menggunakan langkah-langkah yang sudah ditentukan pada algoritma runut-balik di atas maka dapat ditemukan:

Kondisi awal kuda berada pada (B,8) seperti yang ditunjukkan pada gambar 7 di atas. Pada tiap kotak pada papan catur, kita akan membuat daftar kotak yang dapat

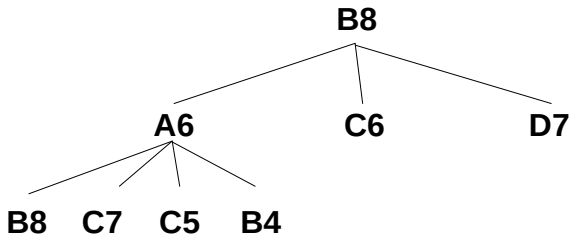
dilalui oleh kuda pada langkah selanjutnya.

Pada kasus ini, ada tiga kemungkinan langkah, yaitu kotak A6, C6, dan D7 yang dalam representasi pohon dapat dilihat pada gambar 9 di bawah ini.



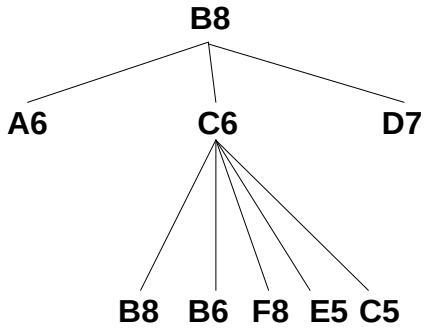
Gambar 9 ilustrasi pohon

Pertama kita kunjungi A6. Jika langkah pada kotak ini belum pernah dikunjungi maka kembangkan anak-anak (opsi-opsi langkah) dari simpul ini, maka kita dapatkan kotak B8, C7, C5, dan B4 seperti yang dapat kita lihat pada gambar 10 di bawah ini.



Gambar 10 ilustrasi pohon pengembangan A6

Lalu kita kunjungi kemungkinan langkah lainnya yaitu C6. Seperti pada langkah sebelumnya, jika kotak ini belum pernah dikunjungi maka kembangkan anak-anak (opsi-opsi langkah) dari simpul ini, maka kita dapatkan kotak B8, C7, C5, dan B4 seperti yang dapat kita lihat pada gambar 10 di bawah ini.



Gambar 11 ilustrasi pohon pengembangan C6

Langkah diulangi seterusnya sampai ditemukan solusi yang diinginkan. Solusi yang diinginkan adalah semua kotak telah dikunjungi.

Algoritma BFS dapat dituliskan dengan *pseudo code* sebagai berikut:

```
Procedure BFS(input v:integer)
{
   Traversal graf dengan algoritma
   pencarian BFS.
   Masukan: v adalah simpul awal kunjungan
   Keluaran: semua simpul yang dikunjungi
   dicetak ke layar
}


Deklarasi
w : integer
q : antrian;

procedure BuatAntrian(input/output q :
antrian)
{ membuat antrian kosong, kepala(q) diisi 0
}

procedure MasukAntrian(input/output
q:antrian, input v:integer)
{ memasukkan v ke dalam antrian q pada
posisi belakang }

procedure HapusAntrian(input/output
q:antrian,output v:integer)
{ menghapus v dari kepala antrian q }

function AntrianKosong(input q:antrian) ®
boolean
{ true jika antrian q kosong, false jika
sebaliknya }


Algoritma:

BuatAntrian(q) { buat antrian kosong }

write(v) { cetak simpul awal yang
dikunjungi }

dikunjungi[v]¬true { simpul v telah
dikunjungi, tandai dengan
true}

MasukAntrian(q,v) { masukkan simpul awal
kunjungan ke dalam
antrian}
{ kunjungi semua simpul graf selama antrian
belum kosong }

while not AntrianKosong(q) do
HapusAntrian(q,v) { simpul v telah
dikunjungi, hapus dari
antrian }
for tiap simpul w yang bertetangga dengan
simpul v do
if not dikunjungi[w] then
write(w) {cetak simpul yang dikunjungi}
MasukAntrian(q,w)
dikunjungi[w]¬true
endif
endfor
endwhile
{ AntrianKosong(q) }
```

VI. KESIMPULAN

Knight's Tour adalah salah satu masalah yang dapat direpresentasikan sebagai graf, atau lebih khususnya pohon, pada permainan catur papan. Ada banyak strategi pencarian algoritmik yang dapat diterapkan untuk menentukan *Knight's Tour* pada permainan tersebut. Salah satunya dapat menggunakan algoritma BFS.

REFERENSI

[1] R. Munir, "Pengantar Strategi Algoritma", http://www.informatika.org/~rinaldi/Stmik/2006-2007/bahan-kuliah2007.htm, waktu akses 9 Desember 2010 21:43.

[2] Lewandowski, Gary. "Project 1: *The Knight's Tour* Gary Lewandowski", CSCI 220, fall 2001

[3] F. Mumtaz, "Aplikasi Teori Graf pada Knight's Tour", submitted for Matematika Diskrit assignment, published at http://www.informatika.org/~rinaldi/Matdis/2007-2008/Makalah-2007.htm.

[4] R. Munir, "Pohon", http://www.informatika.org/~rinaldi/Matdis/2009-2010/strukdis09-10.htm, waktu akses 9 Desember 2010 21:43.

[5] http://en.wikipedia.org/wiki/Breadth-first_search, waktu akses 9 Desember 2010 22:26.

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, 29 April 2010

ttd

Fahmi Mumtaz
13506045