Implementasi Algoritma DFS pada permainan Monument Valley

Muhammad Aodyra Khaidir and 13513063[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*aodyra@itb.ac.id*

***Abstract*—Monument Valley merupakan permainan *puzzle game* yang meminta anda untuk membawa seorang putri yang melalui jalur yang benar ke sebuah pintu. Ada beberapa algoritma yang dapat digunakan untuk menetukan jalur yang benar, yaitu Depth First Search, Breadth First Search, Branch and Bound. Algoritma Depth First Search (DFS) merupakan algoritma pencarian mendalam yang dapat digunakan untuk menentukan jalur yang benar untuk membawa sang putri.**

Fitur penting yang membuat permainan ini menarik adalah karena perspectif yang berubah – ubah dalam menelusuri jalur. Makalah ini membahas penerapan algoritma DFS, khususnya dalam penentuan jalur yang benar.

***Kata Kunci*—monument valley, DFS, jalur benar.**

I. PENDAHULUAN

Monument valley merupakan permainan dengan jenis *puzzle game*, sehingga mengaharuskan pemakainya untuk meyelesaikan masalah yang disediakan dalam game tersebut. Permainan ini tidak banyak memberikan fitur, namun yang membuatnya menarik adalah sudut pandang dari setiap jalur yang menarik.

[2]Monument Valley merupakan permainan berbasis mobile device. Permainan ini cukup popular di Play Store dan App Store. Permainan ini meminta *user* atau pengguna untuk mengantarkan seorang putri untuk sampai ketujuannya. Pada permainan ini, terdapat banyak level dimana setiap level memiliki jalur yang berbeda untuk dilewati putri tersebut.

Permainan ini memberikan berbagai macam pandagan, sehingga terdapat jalur - jalur yang tidak disangka pengguna. Permainan ini memberikan efek tiga dimens pada setiap jalurnya, sehingga suasana yang deberikan oleh game ini lebih terasa oleh pengguna.

Di permainan ini terdapat tombol atau pijakan – pijakan untuk sang putri yang merubah perspektif atau membuka jalur rahasia yang bias dilewati sang putri untuk sampai ke tempat tujuannya. User diminta untuk menggerakkan putri tersebut untuk membuka jalur – jalur rahasia yang ada, sehingga tempat tujuan atau jalut menuju tempat tujuan terbuka.

Pada setiap level dalam game ini memiliki tingkat kesulitan dari perspektif jalur yang berbeda - beda, semakin tinggai levelnya perspektif dari jalur semakin sulit untuk terlihat, sehingga user diminta untuk menggunakan imajinasinya agar jalur tersembunyi dapat terlihat.

Algortima DFS merupakan algoritma pencarian mendalam, sehingga jalur – jalur yang harus menggunakan perspektif tertentu bisa terbaca.

Gambar 1: Permainan Monument Valley yang sedang berjalan
Sumber:https://play.google.com/store/apps/details?id=com.ustwo.monumentvalley

Seperti banyak permainan logika penentuan jalur, banyak jalur yanga dapat dilewati, namu hanya jalur yang sampai ketujuan yang benar. Dalam game ini, jika jalur yang dilewati salah, maka user harus kemabli atau

mengambil jalur lain.

Pada gambar 1, terlihat suatu tombol. Tombol ini akan merubah jalur atau perspektif sehingga dalam penentuan solusi setiap perspektif atau jalur berubah makan solusi akan dicari berdasarkan jalur atau perspektif yang saat ini terlihat.

Dari penjelasan diatas, diketahui bahwa jalur dapat berubah, sesuai perspektif yang terlihat. Sehingga, diperlukan algoritma yang selalu berjalan jika terjadi perubahan jalur atau perspektif. Implementasi algoritma DFS dapat dilakukan secara rekursif, sehingga pada setiap simpul yang ditelusuri algoritma DFS bekerja.

II. DASAR TEORI

Untuk algoritma traversal graf terdapat beberapa algoritma yang mengunjungi simpul secara sistematik[1]:

- Pencarian melebar
  Brearh First Search (BFS).
- Pencarian mendalam
  Depth First Search (DFS).

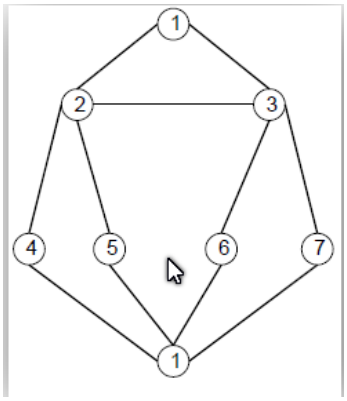
Kedua algoritma merupakan algoritma blind search atau tanpa informasi. Tidak ada informasi tambahan dalam algoritma ini. Dalam makalah ini kita akan memperdalam teori tentang Depth First Search atau disingkat DFS.

Dalam pencarian solusi, terdapat dua pendekatan:

1. Graf Statis: graf yang sudah terbentuk sebelum proses pencarian dilakukan
2. Graf Dinamis: graf yang terbentuk saat proses pencarian dilakukan.

[3]DFS merupakan algoritma pencarian traversal graf yang mengunjungi sebuah simpul awal, dilanjutkan dengan mengunjungi simpul – simpul tetangganya secra mendalam lalu ulangi DFS mulai dari simpul tetangga tersebut. Jika pencarian pada suatu simpul sudah terbatas dan tidak menemukan solusi yang dicari, maka pencarian dirunut balik ke simpul terakhir yang dikunjungi sebelumnya. Pencarian ini dilakukan secara rekursif pada simpul – simpul tetangganya.

Berikut adalah gambar ilustrasi pencarian DFS dengan simpul – simpul yang dikunjungi sesuai dengan nomor simpul:



Gambar 2. Ilustrasi algoritma DFS

Sumber: Diktat Kuliah IF2251 Strategi Algoritmik

Algoritma DFS dapat digambarkan seperti dibawah, dengan memisalkan simpul awal adalah simpul v:

1. Kunjungi simpul v
2. Kunjungi simpul w yang betetangga dengan simpul v
3. Ulangi DFS mulai dari simpul w
4. Ketika mencapai simpul u sedemikian sehingga semua simpul yang bertetangga dengannya dikunjungi, pencarian runut balik (backtrack) ke simpul terakhir yng dikunjungi seblumnya dan mempunyai simpul w yang belum dikunjungi.
5. Pencarian berakhir bila tidak ada lagi simpul yang belum dikunjungi yang dapat dicapai dari simpul yang telah dikunjungi.

Untuk memperjelas algortima yang digunakan, lihat pada gambar 2. Pada gambar 2, penelusuran dimulai dari simpul 1. Setalah itu, telususri simpul 2 yang merupakan anak dari simpul 1. Dari simpul 2, telusuri simpul 3 yang merupakan tetangga dari simpul 2. Dari 3 kita mengunjungi simpul 6 yange merupakan anak pertama dari simpul 3. Dari simpul 6 kunjungi simpul 8. Setelah itu kungjungi simpul 4 simpul tetangga dari simpul 8. Karena simpul 4 tidak memiliki teteangga yng belum diknjungi maka lakukan backtracking ke simpul sebelumnya, dilanjutkan ke simpul 5. Sama seperti simpul 4, simpul 5 tidak memiliki simpul tetangga yang belum dikunjungi, sehingga melakukan backtracking ke simpul selanjutnya dan mengunjungi simpul 7. Jadi secara umumurutan simpul yang dikunjungi adalah 1,2,3,6,8,4,5,7.

Algoritma DFS dapat diimplentasi secara rekursif ditunjukkan dengan pseudocode berikut.

```
procedure DFS(input v:integer)
{
Mengunjungi seluruh simpul graf dengan algoritma
pencarian DFS

Masukan: v adalah simpul awal kunjungan
Keluaran: semua simpulyang dikunjungi ditulis ke layar
}
Deklarasi
    w : integer
Algoritma:
    write(v)
    dikunjungi[v]<–true
    for w<–l to n do
        if A[v,w]=l then {simpul v dan simpul w bertetangga}
                if not dikunjungi[w] then
                        DFS(w)
                endif
        endif
endfor
```

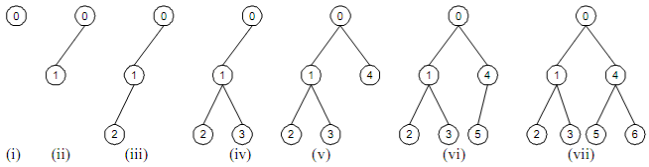
[5]Algoritma DFS bisa menyelesaikan persoalan dengan pencarian. Dalam pencarian solusi, dibentuk pohon dinamis sampai solusi ditemukan dengan metode

traversal DFS atau BFS. Pada setiap simpul diperiksa apakah solusi telah dicapai atau tidak. Jika simpul solusi sudah ketemu, pencarian dapat dihentikan atau mencari solusi lain.

Pohon dinamis dapat direpresentasikan sebagai berikut:

- Pohon ruang status
- Simpul: problem state layak membentuk solusi
  - Akar: initial state
  - Daun: solution / goal state
- Cabang: operator / langkah dalam persoalan
- Ruang status (state space): himpunan semua simpul
- Ruang solusi: himpunan status solusi

Dibawah ini gambar pembentukan pohon dinamis



Gambar 3. Proses pembentukan pohon dinamis secara DFS
Sumber: Sumber: Diktat Kuliah IF2251 Strategi Algoritmik

Pada gambar 3.i pada awal proses terdapat satu simpul yang merupakan simpul awal. simpul selanjutnya dibangkkitkan menjadi simpul anak pertama dari simpul awal (gambar 3.ii). algoritma DFS diterapkan kembali pada simpul ini, sehinggal simpul selanjutnya yang dibngkitkan merupakan anak dari simpul ini. Jika simpul sudah pada batasnya, simpul yang selanjutnya dibangkitkan adalah simpul lain yang berhubungan dengan simpul bapak, namun belum pernah dikunjungi. Hal ini berlanjut sampai semua simpul dikunjungi atau solusi sudah ditemukan. Pada gambar 3.vii diperlihatkan semua simpul sudah dikunjungi.

Kopleksitas waktu algoritma DFS pasa kasus terburuk adalah $O(b^m)$. Kompleksitas ruangalgoritma DFS adalah $O(bm)$, karena kita hanya perlu menyimpan satu buah lintasan tunggal dari akar sampai daun, ditambah dengan simpul – simpul yang terhubung dan belum dikembangkan.

III. Implementasi Algoritma DFS

A. Implementasi Secara Umum

Algoritma DFS akan menelusuri graf sampai kedalaman tertentu, baru setelah tidak memiliki simpul lain yang belum dikunjungi backtracking ke simpul yang memiliki anak yang belum dikunjungi. Hingga putri sampai pada tujuan.

Secara umum, implementasi yang digunakan, pertama telusuri jalur dari posisi putri sekarang ke jalur – jalur yang tersedia. Jika jalur tersebut memiliki anak jalan lain kunjungi jalur tersebut dan berhenti jika jalur terputus atau sudah menemukan solusi.

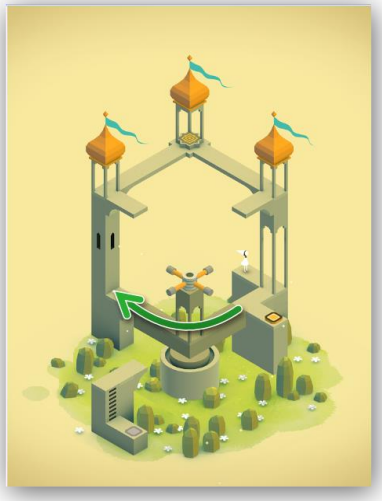
B. Pencarian Solusi dengan Algoritma DFS

Dalam pencarian solusi menggunakan algoritma DFS, dibentuk pohoh dinamis. Disini simpul awal merupakan posisi putri saat ini. Ketika bertemu di persimpangan yang yang pecah manjadi dua atau lebih jalur, simpul awal tersebut memiliki dua anak atau lebih

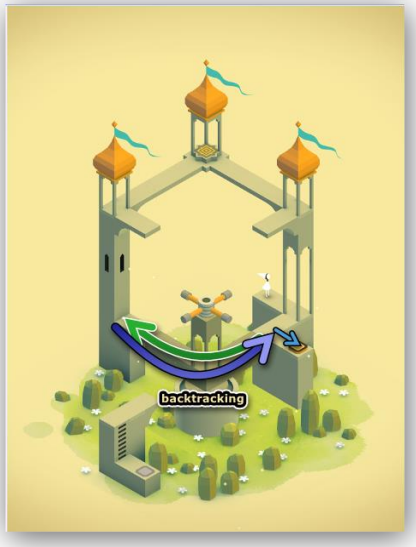
Gambar 3. Ilustrasi pembentukan simpul pada game
Sumber:https://play.google.com/store/apps/details?id=com.ustwo.monumentvalley

Seperti yang diperlihatkan pada gambar 3, simpul awal adalah posisi putri saat ini, jika ketika melewati jalur menemukan sebuah persimpangan yang bercabang menjadi dua atau lebih cabang, maka terbentuk simpul anak yang merupakan salah satu dari cabang tertentu. Pada cabang tersebut atau simpul anak pertama dari simpul awal lakukan algortima DFS kembali, proses rekursif ini berhenti jika simpul anak tidak memiliki cabang lagi dan bukan merupakan solusi.

Pada gambar 4 dibawah ditampilkan penelusuran pada jalur hijau yang dilakukan putri dengan mengambil simpul anak tertentu. Di ujing jalur terlihat jalan yang buntu, sehingga putri tidak bisa kemana – mana lagi. Jika digambarkan grafnya simpul yang bersesuaian adalah sebuah simpul yang tidak memiliki simpul yang terhubung ke simpul lain. Jika terjadi hal seperti ini, maka putri harus kembali ke simpul sebelumnya (backtracking) dan melanjutkan ke simpul yang belum dikunjungi. Seperti gambar 5, putri yang melewati jalur hijau akan menemukan jalur buntu sehinga melakukan backtracking melewati jalur biru ke simpulnya. Setelah sampai ke simpul sebelumnya dilanjutkan ke simpul anak yang belum di kunjungi, yaitu jalur biru muda.



Gambar 4. Ilustrasi pembentukan simpul pada game
Sumber:https://play.google.com/store/apps/details?id=com.ustwo.monumentvalley



Gambar 5. Backtracking ke simpul sebelumnya
Sumber:https://play.google.com/store/apps/details?id=com.ustwo.monumentvalley

Hal ini dilakukan secara berulang sampai putri sudah menemukan solusi atau sudah mengunjungi semua simpul namun tidsak menemukan solusi. Jalur yang ditelusuri putri dilakukan secara DFS, sehingga putri akan selalu menelusuri jalan sampai jalan buntu dan melakukan backtracking.

Perubahan perspektif dapat terjadi jika suatu tombol Tertekan oleh putri. Pada saat ini algoritma DFS dilakukan kembali dengan jalu baru. Sehingga simpul awal ditentukan kembali, yaitu posisi putri saat ini.

- Tentukan simpul awal yang baru, yaitu posisi putri saat ini
- Ikuti jalur sampai ke persimpangan yang bercabang, cabang – cabang yang ada merupakan anak dari simpul awal.
- lakukan algoritma DFS pada simpul anak pertama
- jika ditemukan simpul solusi, maka selesai. Jika bertemu jalan buntu putri harus kembali ke simpul sebelumnya atau backtracking untuk ke simpul selanjutnya yang belum dikunjungi.
- Lalu lakukan algoritma kambali pada simpul yang sedang dikunjungi.
- Proses ini berakhir jika semua simpul sudah dikunjungi atau sudah menemukan solusi

Gambar 6. Perubahan perspektif
Sumber:https://play.google.com/store/apps/details?id=com.ustwo.monumentvalley

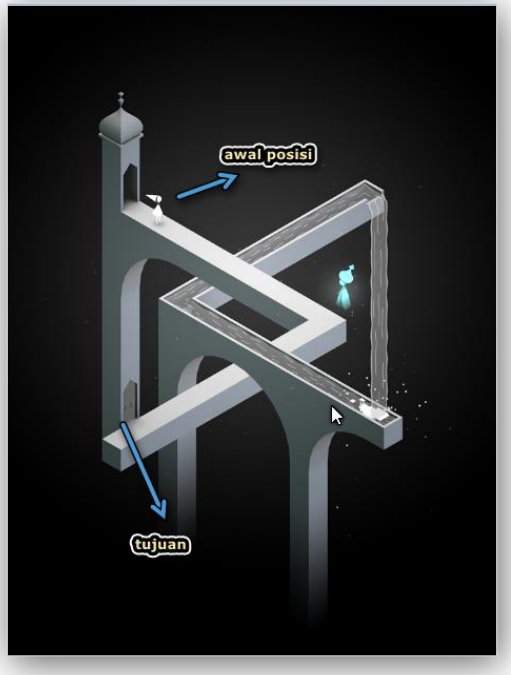
Berikut adalah pseudo code tentang implemntasi DFS pada permainan Monument Valley:

```
Procedure DFS(input v:integer)
{
        Mengunjungi seluruh jalur dengan pecncaria
DFS.
}
Deklarasi
    w: integer
Algoritma:
    Dikunjungi[v] <- true
    While (putriTidakSampai())
        If (cabangJalan berhubungan jalanDitelusuri)then
                If jalanbelumdikunjungi(w) then
                        DFS(w)
                endif
        endif
    endwhile
```

Dari pseudo conde diatas , diperlihatkan jika putri belum sampai maka proses akan terus berlanjut sampai putri sampai pada tujuan. Cabang jalan sama separti simpul terhubung lain yang sudah atau belum dinkunjungi.

Jika user berhasil mengantarkan putri, makan user akan masuk ke level selanjutnya yang lebih sulit dibandingkan level sebelumnya. Walaupun lebih sulit, namun tujuannya teteap sama yaitu mengantarkan putri ketujuan. Sama jika uer tidak berhasil mengantarkan sang putri samapi ke tempat tujuan, maka user akan kembali ke jalur sebelumnya, untuk mengulangi proses DFS yang sama dalam jalur yang berbeda atau belum pernah dikunjungi.

Fungsi tombol peubah perspektif atau peubah jalur adalah sebagai tanda, dimulainya proses algoritma DFS kembali dari titik tersebut menuju tempat tujuan.



Gambar 7. Permainan yang sedang berjalan
Sumber:https://play.google.com/store/apps/details?id=com.ustwo.monumentvalley

Dalam menentukan jalur yang benar, setelah kita mempunya simpul awal, kita akan mencari jalur ke tujuan yaitu sebuah pintu. Seperti terlihar pada gambar 7.ada daerah tertutup pada jalur yang tidak terlihat. Agar sang putri sampai ketujuan dengan cepat maka kita harus menentukan jalur mana yang benar. Walaupun jalur terlihat terpotong, karena permainan ini merupakan permainan dengan grefis 3D, maka jalur tetap ada.

Pada gambar 7 tidak ada percabangan, sehingga dari simpul awal tidak ada simpul lain yang terhubung. Kalau simpulnya hanya terdapat satu simpul.

Gambar 8. Simpul yang terbentuk pada pohon dinamis

Pada gambar 8. Terlihat hanya satu simpul yang dihasilkan dari pohon dinamis. Simpul ini merupakan simpul daun yang berupa solusi dari permainan ini. setelah game

IV. KESIMPULAN DAN SARAN

Monument Valley merupakan salah satu game puzzle yang bertujuan mencari jalur yang benar ke sebuah tujuan. Pada game ini ada beberapa fitur menarik, yaitu sudut pandang atai perspekti yang dapat berubah – ubah serta jalur rahasia yang mukin tersembunyi karena perspektif itu. Kita diminta untuk mengarahkan seorang putri sampai ketempat tujuannya. Namun hal ini tidak mudah karena jalur yang sulit dilewati karena perspektif – perspektif sehingga beberapa jalan tertutup.

Algoritma DFS adalah sebuah algoritma pencarian traversal graf yang bisa digunakan dalam pencarian solusi. Algoritma ini berbeda dari algoritma traversal graf lain, terutama algoritma ini melakukan pencarian secara mendalam keanak – anaknya dan dilanjutkan dengan simpul – simpul terhubung lain yang belum ditelusuri dan dikembangkan. Dalam pencarian solusi, pohon dinamis terbentuk sesuai dengan algoritma yang digunakan. Saat kita menggunakan algoritma DFS, maka pohon dinamik akan terbentuk mendalam.

Dalam permainan ini, dibutuhkan algoritma yang memudahkan user untuk mengetahui jalur yang benar. Salah satu algoritma yang cocok dengan masalah ini adalah DFS yang diimplementasikan pada sebuah percabangan. Algoritma ini akan menelusuri setiap cabang secara mendalam dan menemukan jalur yang benar untuk sampai ke tujuan. Jika pada cabang tertentu merupakan jalur buntu, maka user akan melakukan backtracking ke simpul sebelumnya dan mengunjungi simpul – simpul tehubung lain yang belum di telusuri dan dikembangkan.

V. ACKNOWLEDGMENT

Puji syukur penulis panjatkan kepada tuhan YME, berkat rahmat dan izin-nya makalah ini dapat diselesaikan tepat waktu. Rasa terima kasih juga penulis ucapkan kepada Dr. Ir. Rinaldi Munir, MT, dan Harlili S., M.Sc. atas bimbingannya selaku dosen matematika diskrit. Penulis juga mengucapkan terima kasih kepada orang tua penulis, dan teman – teman atas segala doa, bantuan, masukan, sehingga masalah ini dapat selesai.

REFERENCES

[1] Munir, Rinaldi. Diktat Kuliah IF2211 Strategi Algoritma. 2009.
[2] http://en.wikipedia.org/wiki/Depth-first_search tangggal akses 4 Mei 2015
[3] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2014-2015/BFS%20dan%20DFS%20(2015).pptx diakses tanggal 4 Mei 2015
[4] http://www.programmerinterview.com/index.php/data-structures/dfs-vs-bfs/ diakses tanggal 4 Mei 2015
[5] https://www.ics.uci.edu/~eppstein/161/960215.html diakses tanggal 4 Mei 2015

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, 5 Mei 2015

Muhammad Aodyra Khaidir
13513063