Aplikasi Algoritma A* dan Algoritma BFS dalam Permainan Daring Ragnarok M: Eternal Love

Winston Wijaya - 13517018
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517018@std.stei.itb.ac.id

***Abstract*—Dalam kehidupan sehari-hari, ada banyak sekali kontribusi algoritma A* dan BFS dalam berbagai aktivitas dan perangkat lunak yang digunakan, salah satunya dalam beberapa permainan daring seperti Ragnarok M: Eternal Love. Pada makalah ini akan membahas kontribusi algoritma A* dan BFS pada permainan dari Ragnarok M: Eternal Love.**

***Keywords*—A*, BFS, Kontribusi, Rute Terpendek**

I. PENDAHULUAN

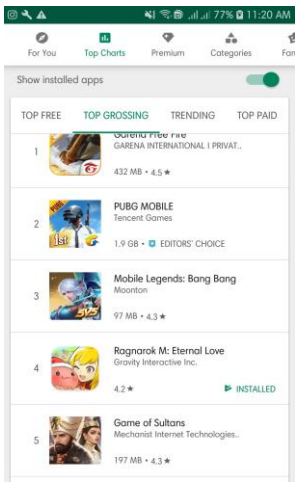
Kian berkembangnya ilmu pengetahuan dan teknologi mendorongan berbagai aktor dalam dunia industri untuk menciptakan kreasi baru yang lebih menarik dan mampu bersaing di masyarakat, salah satunya dalam industri permainan. Pada era modern ini sudah banyak permainan yang dimainkan oleh masyarakat dari berbagai kalangan, salah satunya permainan daring yang telah dikemas dalam berbagai *genre*.

Secara sempit, permainan daring dapat diartikan sebagai *video game* yang dimainkan oleh pemarin dari berbagai tempat yang terhubung oleh jaringan komputer seperti internet atau teknologi yang sebanding[1]. Permainan daring sangat bervariasi, mulai dari yang hanya berbasis teks hingga permainan yang melibatkan grafik yang sangat kompleks dan dapat dimainkan secara bersama oleh banyak pemain dari berbagai tempat dalam waktu yang bersamaan[1]. Banyaknya pemain dari berbagai tempat pun mendorong terbentuknya berbagai perkumpulan atau komunitas yang menjadi wadah untuk berbagai cerita, saran, dan pengalaman berkaitan dengan permainan tersebut.

Makin berkembangnya teknologi memacu para *game developer* dan perusahaan dalam industri permain untuk menciptakan permainan yang menarik minat para pemain, dapat bertahan lama, dan mampu bersaing di pasar, salah satunya adalah GRAVITY Co., Ltd yang merilis *game* Ragnarok M: Eternal Love pada tahun 2018. *Game* tersebut sempat menduduki peringkat pertama sebagai *The Highest Grossing Ranking* pada *Google Play* dan *Apple App Store* dan sekarang masih menduduki peringkat keempat pada kategori permainan *Top Grossing* pada *Google Play Store*.

The Highest Grossing Ranking in Southeast Asia (as of 5 November 2018)		
	Google play	Apple App Store
Thailand	No. 1	No. 1
The Philippines	No. 1	No. 1
Indonesia	No. 1	No. 1

Gambar 1.1 Peringkat Ragnarok M: Eternal Love per 5 November 2018[2]



Gambar 1.2 Peringkat Ragnarok M: Eternal Love per 26 April 2019

Game ini tergolong game *mobile* MMORPG yang unik. Dalam permainannya mulai diterpkan beberapa fitur yang umumnya dijumpai pada *game* yang berbasis *Personal Computer* (PC).

II. DASAR TEORI

*2.1 Algoritma A**

Algoritma A* merupaka sebuah algoritma yang digunakan untuk melakukan pencarian rute terpendek dari suatu titik awal menuju titik akhir dengan menghindari pembangkitan kemungkinan yang memiliki biaya yang sudah berlebih.

Pada algoritma ini terdapat sebuah fungsi evaluasi, misalnya f(n) dengan persamaan sebagai berikut[2]:

$$f(n) = g(n) + h(n)$$

f(n) = estimasi total *cost* dari jalur yang ditempuh melewati

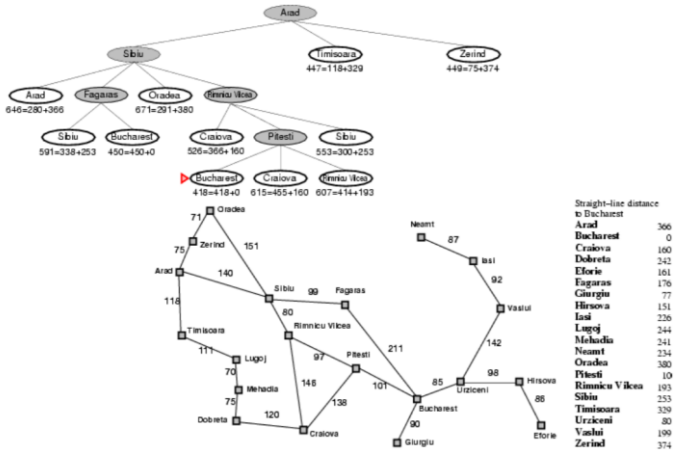
n ke *goal*

g(n) = *cost* yang telah ditempuh untuk sampai ke n

h(n) = estimasi *cost* dari n ke *goal*

Penyelesaian dengan algoritma A* dimulai dari suatu titik, misal *i*. Dari posisi *i* dibangkitkan semua rute yang mungkin kemudian dihitung masing-masing *f(n)* dari rute yang dibangkitkan kemudian dimasukkan ke dalam sebuah *priority queue* yang terurut berdasarkan *cost* terkecil. Pembangkitan berikutnya dilakukan dari rute yang memiliki *cost* terkecil kemudian dimasukkan ke dalam *priority queue* pula yang terurut berdasarkan *cost*. Langkah tersebut diulangi sampai ditemukan *goal* yang diinginkan.

Kemudian ketika sudah ditemukan *goal* yang diinginkan, keluarka semua rute yang memiliki *cost* >= *cost* yang dimiliki rute yang telah mencapai *goal*. Bila masih ada rute yang tersisa lakukan pembangkitan ulang pada rute tersebut hingga tidak ada lagi rute yang dapat dibangkitkan. Hasil yang masih terdapat dalam *priority queuei* yang dibuat sebelumnya adalah



semua rute dengan *cost* terkecil dari titik awal (*i*) menuju *goal*.

Contoh satu contoh persoalan yang paling sering diselesaikan dengan algoritma A* adalah persoalan pencarian rute terpendek dari suatu kota ke kota lainnya dengan contoh sebagai berikut:

Gambar 2.1.1 Contoh Pencarian A* untuk Menentukan Rute Terpendek dari Kota Arad menuju kota Bucharest[3]

Pada contoh di atas g(n) yang digunakan adalah jarak yang ditempuh hingga kota ke-n dan h(n) yang digunakan adalah jarak kartesian dari kota ke-n menuju *goal* dan didapatkan hasil pengerjaan seperti di atas.

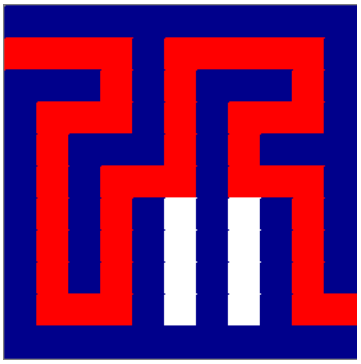
2.2 Algoritma Breadth First Search (BFS)

Algoritma BFS merupakan algoritma *traversal* pada sebuah graf untuk mengunjungi semua simpul pada graf dengan melakukan pencarian melebar[4]. Dalam persoalan ini algoritma BFS dihentikan ketika ditemukan salah satu solusi yang menuju *goal* yang diinginkan.

Pada pecarian rute terpendek dengan algoritma BFS terdapat langkah-langkah penyelesaian sebagai berikut[4]:

1. Kunjungi sebuah simpul, misal *v*.
2. Kunjungi semua simpul yang bertetangga dengan simpul *v* terlebih dahulu.
3. Kunjungi simpul yang belum dikunjungi dan bertetanggaan dengan simpul-simpul yang tadi dikunjungi, demikian seterusnya hingga ditemui simpul *goal* yang diinginkan.

Salah satu penyelesaian rute terpendek dengan menggunakan algoritma BFS adalah penentuan rute terpendek untuk menemukan jalan keluar dari labirin.

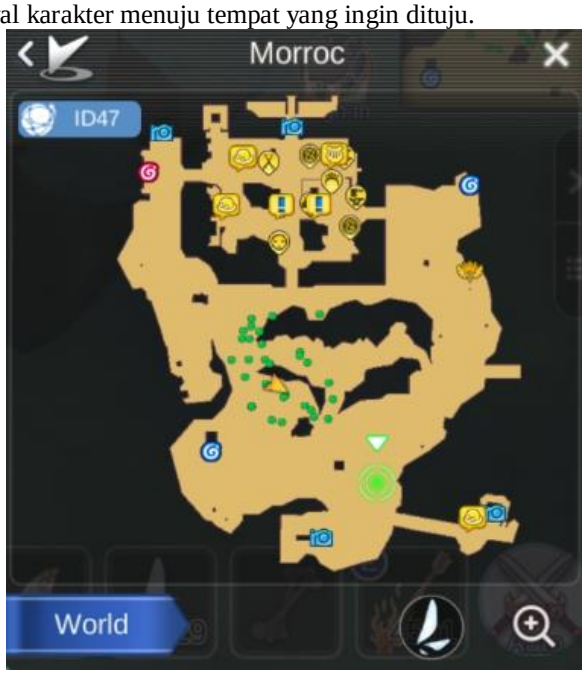


Gambar 2.2.1 Contoh Penyelesaian Labirin dengan Algoritma BFS

III. KONTRIBUSI ALGORITMA A* DAN BFS PADA PERMAINAN DARING RAGNAROK M: ETERNAL LOVE

3.1 Fitur *Auto-Move*

Fitur ini merupakan salah satu fitur yang memudahkan pemain untuk memindahkan karakternya dari suatu posisi pada peta menuju posisi lain pada peta tersebut yang biasanya dimanfaatkan untuk menuju sebuat *Non Player Character* (NPC) atau *portal* antar peta. Cara menggunakan fitur ini adalah dengan mengklik suatu titik pada peta dan sistem pada *game* akan secara otomatis menggerakkan player dari posisi awal karakter menuju tempat yang ingin dituju.



Gambar 3.1.1 Peta Morroc pada Ragnarok M: Eternal Love

Pada contoh di atas, terlihat sebuah panah berwarna kuning uang menunu posisi karakter sekarang (*current position*) menuju posisi akhir yang diinginkan. Fitur *auto-move* akan menentukan rute terpendek yang memungkinkan dari posisi karakter sekarang menuju posisi akhir dan setiap satu perpindahan karakter, akan ada penentuan ulang rute terpendek dari posisi karakter pemain yang baru menuju posisi akhir yang diinginkan.

Algoritma BFS yang berkontribusi untuk menentukan rute terpendek pada fitur *auto-move* dalam notasi algoritma dengan input 2 buah *tuple*, yakni *tuple* posisi awal dan *tuple* posisi akhir sebagai berikut:

```
funtion BFS (input start: (integer,integer), input finisih:
(integer, integer) ) → antrian
{
  Melakukan traversal dari tuple start hingga menncapai
  Tuple finish menggunakan algoritma BFS

 Input: tuple start yang meruapakan titik awal player dan
        Tuple finish yang merupakan titik tujuan player
 Output: sebuah rute yang diharapkan jalur terpendek
         dari tuple start menuju tuple finish
}
Deklarasi
   q : antrian
   pt : (integer, integer)
   pt_temp : (integer, integer)
   sol : antrian

  procedure createQueue(input/output q : antrian)
 {membuat antrian kosong}

  function isEmpty(input q : antrian) → boolean
 {mengecek apakah antrian kosong}

  procedure inputTuple(input/output q : antrian, input
                       point : (integer, integer))
 {memasukkan point ke dalam antrian}

  function getPoint(input q : antrian) → (integer, integer)
 {mengembalikan sebuah tuple titik pada awal queue}

  function findRoute(input q : antrian, input target :
                     (integer, integer) → antrian
 {menetukan rute yang dilewati}

Algoritma
  createQueue(q)            {membuat antrian kosong}
  inputTuple(q,start)   {memasukkan start dalam antrian}
  dikunjungi{start} → true

  while not(isEmpty(q)) do
     pt ← getPoint(q)
    for semua tetangga pt_temp dari titik pt do
       dikunjungi{pt_temp}
       inputTuple(q,pt_temp)
    endfor
  endwhile
  sol ← findRoute(q,finish)
  ← sol
```

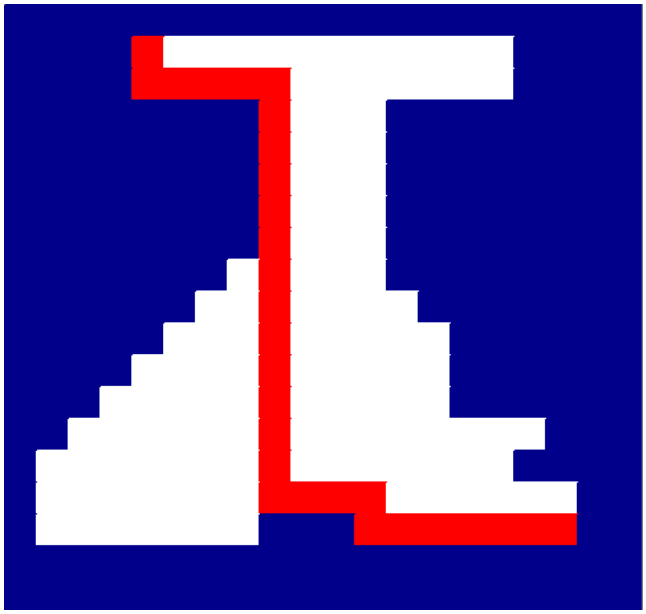
Berikut adalah algoritma *auto-move* yang membangkitkan algoritma BFS:

```
procedure AutoMove(input start: (integer,integer),
input finisih: (integer, integer) )
{
  Melakukan perpindahan otomatis pemain dari titik
  start menuju titik finish

  Input: tuple start yang meruapakan titik awal player
         dan Tuple finish yang merupakan titik tujuan
         player
  Output: Player berada di titik yang diinginkan
}
Deklarasi
   q : antrian
   pt : (integer, integer)

   function isPointEqual(input p1 : (integer, integer),
                         input p2 : (integer, integer)) → boolean
  {mengecek apakah dua buah titik sama}

  function getPoint(input q : antrian) → (integer, integer)
  {mengembalikan sebuah tuple titik pada awal queue}

Algoritma
      if isPointEqual(start,finish) then
          {do nothing → sudah sampai tujuan}
      else
          q ← BFS(start,finish)
          pt ← getPoint(q)
          AutoMove(pt,finish)
```

Dengan algoritma di atas, maka dapat ditemukan rute terpendek yang optimal dari titik awal hingga titik akhir di tiap perpindahan karakter.

Berikut adalah contoh peta dan hasil dari salah satu rute BFS yang diperoleh:

Gamba 3.1.2 Peta Prontera



Gambar 3.1.3 Hasil Algoritma BFS

3.2 Fitur Rune

Fitur rune merupakan salah satu fitur yang favorit bagi para pemain terutama pemain yang sudah ahli yang tergabung dalam *guild* untuk memperkuat karakter mereka. Fitur ini sendiri digambarkan sebagai sebuah graf berobot dengan 2 jenis bobot, yakni jumlah kontribusi dan jumlah *gold medal*.

Untuk membantu player, developer sudah menyediakan fitur untum memberikan saran rute yang harus ditempuh untuk mengaktifkan rune dari posisi rune aktif terdekat

Dalam fitur ini terdapat:

$$f(n) = g(n) + h(n)$$

Dengan:

f(n) = estimasi total *cost* dari jalur yang ditempuh melewati
g(n) = total *gold medal* dan kontribusi yang telah dipakai
h(n) = estimasi *cost gold medal* dan kontribusi dari n ke *goal*

Dalam *g(n)* sendiri jumlah *gold medal* memiliki penilaian yang lebih tinggi dibandingkan jumlah kontribusi, sehingga walaupun ada jalur dengan kontribusi yang lebih keci, jalur dengan *gold medal* yang lebih kecil dipilih dibandingkan dengan jalur dengan kontibusi lebih kecil.

IV. Kesimpulan

Dari penjelasan yang telah dipaparkan di atas, dapat ditarik kesimpulan bahwa dalam pembuatan fitur *auto-move* memerlukan algoritma yang mampu mengoptimasi langkah yang diperlukan ke *goal* dan dengan waktu eksekusi yang lebih kecil menggunakan berbagai strategi algoritma yang mungkin, salah satunya menggunakan BFS. Sama halnya dengan fitur *auto-move*, sistem rune juga memerlukan algoritma seperti A* untuk menentukan jalur yang terpendek dan memberikan saran kepada pemain.

Acknowledgment

Pertama-tama penulis mengucapkan syukur atas kehadirat Tuhan Yang Maha Esa, atas berkat dan karunianya saya mampu menyelesaikan makalah ini dan menempuh studi di Institut Teknologi Bandung.

Selanjutnya, penulis juga mengucapkan terima kasih kepada orang tua dan keluarga atas dukungan dan doa yang senantiasa diberikan sehingga penulis mampu menyelesaikan makalah dan studi saat ini.

Tidak lupa pula penulis juga mengucapkan terima kasih kepada Bapak Rinaldi Munir selaku dosen Strategi Algoritma kelas K03 tahun 2019 atas limpahan ilmu dan bimbingannya yang sangat bermanfaat dalam pembuatn makalah ini.

Terakhir, penulis juga tidak lupa untuk mengucapkan terima kasih kepada teman-teman atas dukungannya sehingga tugas ini dapat diselesaikan

References

[1] https://www.definitions.net/definition/online+game (Diakses terakhir pada tanggal 25 April 2019)

[2] https://www.globenewswire.com/news-release/2018/11/06/1645929/0/en/Ragnarok-M-Eternal-Love-marks-No-1-on-Google-Play-in-Three-Countries-followed-by-Apple-Store.html (Diakses terakhir pada tanggal 26 April 2019)

[3] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/A-Star-Best-FS-dan-UCS-(2018).pdf (Diakses terakhir pada tanggal 26 April 2019)

[4] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/BFS-dan-DFS-(2019).pdf (Diakses terakhir pada tanggal 26 April 2019)

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, 26 April 2019

Winston Wijaya (13517018)