Penerapan Algoritma *Greedy* dalam Perancangan Rute *Farming* di Mobile Legends

Gde Anantha Priharsena / 13519026[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13519026@std.stei.itb.ac.id

***Abstrak*—Permainan daring semakin populer seiring dengan perkembangan teknologi. Berbagai jenis permainan daring telah dibuat dan *Multiplayer Online Battle Arena (MOBA)* merupakan salah satu jenis permainan daring yang cukup populer di kalangan masyarakat dunia tidak terkecuali di Indonesia. Mobile Legends Bang Bang yang berbasis MOBA turut mewarnai industri permainan daring dan berhasil memikat hati masyarakat Indonesia karena tidak memerlukan spesifikasi perangkat (telepon pintar) yang cukup tinggi serta memiliki tingkat kesulitan yang cukup rendah. Seperti permainan MOBA pada umumnya, terdapat sebuah istilah yang disebut *farming*. *Farming* adalah mekanisme untuk mendapatkan *gold* dan *exp* sebanyak-banyaknya dengan cara membunuh monster hutan yang tersebar di arena permainan. Oleh karena itu, diperlukan suatu perancangan rute *farming* khususnya untuk pemain yang bertugas sebagai *jungler* dalam sebuah tim. Algoritma *Greedy* dapat dimanfaatkan untuk membuat sebuah rute *farming* di awal permainan yang mencakup keseluruhan monster yang ada di arena permainan dengan keuntungan terbesar sehingga *farming* menjadi lebih efektif dan efisien.**

***Kata Kunci*— Permainan, Daring, Mobile Legends, Greedy**

I. PENDAHULUAN

Perkembangan teknologi semakin meningkat setiap harinya. Industri hiburan merupakan salah satu bidang yang sangat terpengaruh dengan perkembangan teknologi tersebut. Tidak hanya film, musik, drama yang menjadi ujung tombak dari industri hiburan, tetapi juga *video game* yang mulai mendapatkan perhatian khusus di bidang ini. *Video Game* yang tadinya hanya dipandang sebelah mata dan hanya dijadikan sebuah hobi sudah menjadi sebuah industri yang sangat besar sekarang. Semua orang baik muda ataupun tua, laki-laki ataupun perempuan sudah bermain *video game* hampir setiap hari. Awalnya, *video game* hanya dapat dimainkan dengan konsol tertentu seperti XBOX, Playstation. dan Nintendo. Akan tetapi, dengan berkembangnya teknologi para pemilik industri *video game* berhasil membuat permainan daring yang memungkinkan para pemainnya bermain bersama kapanpun dan dimanapun asalkan terhubung dengan koneksi internet. Inovasi ini berhasil membuat industri *video game* semakin besar dan menciptakan komunitas-komunitas baru yang menjadi bagian dari masyarakat.

Permainan daring terus berkembang dan terbagi menjadi beberapa jenis yang memiliki perbedaan karakteristik dan cara memainkannya. Beberapa jenis permainan daring yang cukup populer di masyarakat antara lain *Massively Multiplayer Online (MMO)*, *Real-Time Strategy (RTS)*, *Role-Playing Games (RPG)*, *First-Person Shooting (FPS)*, dan *Multiplayer Online Battle Arena (MOBA)*. *Massively Multiplayer Online (MMO)* adalah permainan daring yang memungkinkan ratusan orang bahkan ribuan orang berada dalam satu server yang sama seperti World of Warcraft dan Eve Online. *Real-Time Strategy (RTS)* merupakan sebuah permainan daring yang membutuhkan keahlian dalam menyusun strategi dimana pemain harus membangun sebuah pasukan, bangunan, dan inventaris seperti Age of Empire. *Role Playing Game (RPG)* adalah jenis permainan daring yang memungkinkan para pemainnya untuk memainkan sebuah peran dalam sebuah cerita seperti Ragnarok Online. *First-Person Shooting (FPS)* adalah jenis permainan daring yang bergenre *shooting* dengan tampilan permainan dari penglihatan mata para pemain seperti Counter Strike dan Point Blank. *Multiplayer Online Battle Arena (MOBA)* adalah sebuah jenis permainan daring yang umumnya dimainkan oleh sepuluh orang dan dibagi menjadi dua tim yang fokus kepada pengembangan skil karakter dan strategi dalam pertempuran antar tim. di dalam arena. Contoh permainan daring jenis ini antara lain DOTA 2 dan Mobile Legends Bang Bang (MLBB).

Diantara kelima jenis permainan daring tersebut, MOBA merupakan salah satu jenis permainan yang sangat digemari oleh masyarakat Indonesia khususnya permainan Mobile Legends Bang Bang (MLBB). Hal ini tentunya sangat beralasan karena MLBB tidak memerlukan spesifikasi perangkat (telepon pintar) yang cukup tinggi serta memiliki tingkat kesulitan yang cukup rendah. Seperti permainan daring jenis MOBA pada umumnya, di MLBB juga terdapat sebuah istilah yang disebut *farming*. *Farming* adalah sebuah mekanisme untuk mendapatkan *gold* dan *exp* sebanyak-banyaknya untuk meningkatkan level dan skil serta membeli barang yang dapat digunakan untuk mengalahkan musuh. *Farming* dilakukan dengan cara membunuh monster hutan yang tersebar di arena permainan. Oleh karena itu, diperlukan suatu perancangan rute *farming* khususnya untuk pemain yang bertugas sebagai *jungler* dalam sebuah tim. Salah satu algoritma yang dapat digunakan dalam menentukan rute *farming* adalah algoritma *Greedy*. Permasalahan ini memiliki konsep yang mirip dengan *Travelling Salesman Problem (TSP)* yang dapat diselesaikan dengan algoritma tersebut. Pemanfaatan algoritma ini bertujuan untuk membuat sebuah rute *farming* di awal permainan yang mencakup keseluruhan monster yang ada di arena permainan

dengan keuntungan terbesar sehingga *farming* menjadi lebih efektif dan efisien.

II. LANDASAN TEORI

A. Algoritma Greedy

Algoritma *Greedy* merupakan metode yang paling populer karena kemudahannya untuk menyelesaikan persoalan optimasi. Persoalan optimasi adalah persoalan yang ditujukan untuk mencari solusi optimal. Terdapat dua jenis persoalan optimasi yaitu maksimasi dan minimasi. Secara definisi, algoritma *Greedy* adalah algoritma yang memecahkan persoalan langkah per langkah (*step by step*) sedemikian sehingga pada setiap langkah:

1. Mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan (prinsip "*take what you can get now!"*).
2. dan berharap bahwa dengan pemilihan optimum lokal pada setiap langkah akan berakhir dengan optimum global.

Algoritma *Greedy* memiliki enam elemen yang digunakan sebagai dasar penentuan solusi. Adapun keenam elemen tersebut antara lain:

1. Himpunan kandidat, C: berisi kandidat yang akan dipilih pada setiap langkah (misal: simpul/sisi di dalam graf, *job*, *task*, koin, benda, karakter, dsb)
2. Himpunan solusi, S: berisi kandidat yang sudah dipilih
3. Fungsi solusi: menentukan apakah himpunan kandidat yang dipilih sudah memberikan solusi
4. Fungsi seleksi (*selection function*): memilih kandidat berdasarkan strategi *greedy* tertentu. Strategi *greedy* ini bersifat heuristik.
5. Fungsi kelayakan (*feasible*): memeriksa apakah kandidat yang dipilih dapat dimasukkan ke dalam himpunan solusi (layak atau tidak)
6. Fungsi obyektif: memaksimumkan atau meminimumkan

Dengan menggunakan elemen-elemen diatas, maka dapat dikatakan bahwa algoritma *greedy* melibatkan pencarian sebuah himpunan bagian, S, dari himpunan kandidat, C; yang dalam hal ini, S harus memenuhi beberapa kriteria yang ditentukan, yaitu S menyatakan suatu solusi dan S dioptimisasi oleh fungsi obyektif. Adapun skema umum algoritma *greedy* sebagai berikut:

```
function greedy(C: himpunan_kandidat) → himpunan_solusi
{Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy}

Deklarasi
x: kandidat
S: himpunan_solusi

Algoritma:
S ← {} {inisialisasi S dengan kosong}
while (not SOLUSI(S)) and (C ≠ {}) do
  x ← SELEKSI(C) { pilih sebuah kandidat dari C}
  C ← C – {x} {buang x dari C karena sudah dipilih}
  if LAYAK(S ∪ {x}) then
  {x memenuhi kelayakan untuk dimasukkan ke dalam himpunan solusi}
    S ← S ∪ {x} {masukkan x ke dalam himpunan solusi}
  endif
endwhile
{SOLUSI(S) or C = {}}

if SOLUSI(S) then {solusi sudah lengkap}
  return S
else
  write('tidak ada solusi')
endif
```

Dari skema tersebut dapat disimpulkan bahwa pada akhir setiap iterasi didapatkan sebuah solusi yang merupakan optimum lokal. Pada akhir kalang **while-do** diperoleh optimum global (jika ada). Akan tetapi, optimum global belum tentu merupakan solusi optimum (terbaik), bisa jadi merupakan solusi *sub-optimum* atau *pseudo-optimum*. Hal ini dikarenakan algorima *greedy* tidak beroperasi secara menyeluruh terhadap semua kemungkinan solusi yang ada. Selain itu, terdapat beberapa fungsi seleksi yang berbeda sehingga kita harus memilih fungsi

Contoh-contoh persoalan yang dapat diselesaikan dengan algoritma *greedy* antara lain persoalan penukaran uang (*coin exchange problem*), persoalan memilih aktivitas (*activity selection problem*), minimisasi waktu dalam sistem, persoalan *knapsack* (*knapsack problem*), penjadwalan *Job* dengan tenggat waktu (*Job scheduling with deadlines*), pohon merantang minimum (*minimum spanning tree*), lintasan terpendek (*shortest path*), kode Huffman (*Huffman code*), pecahan Mesir (*Egyptian Fraction*), dan lain-lain.

B. Travelling Salesman Problem

Travelling Salesman Problem adalah persoalan optimisasi untuk mencari rute perjalanan terpendek bagi pedagang keliling yang ingin mengunjungin beberapa kota dan kembali ke kota asal keberangkatan. Sebetulnya, algoritma *Greedy* bukanlah sebuah metode mutlak untuk menyelesaikan persoalan ini. Terdapat beberapa algoritma lain untuk menyelesaikan persoalan ini seperti *Brute Force, Branch and Bound,* dan *Dynamic Progamming*. Akan tetapi, pada kasus ini akan digunakan algoritma *Greedy* untuk menyelesaikan persoalan *Travelling Salesman Problem*.

Dengan algoritma *greedy* maka dapat ditentukan sebuah jalur terpendek antara simpul-simpul yang akan digunakan dengan mengambil secara terus-menerus dan menambahkannya ke jalur yang akan dilewati. Mengacu pada konsep algoritma *Greedy* menganggap bahwa pada setiap langkah akan dipilih tempat atau kota yang belum pernah dikunjungi, dimana tempat atau kota tersebut memiliki jarak terdekat dari tempat atau kota sebelumnya. Sehingga dengan kata lain, dapat dikatakan bahwa langkah dari algoritma greedy ini adalah mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan, atau dengan prinsip "take what you can get now", berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global. Dengan prinsip seperti ini dapat dikatakan bahwa algoritma greedy lebih berguna untuk menghasilkan solusi hampiran (aproksimasi). Hal ini dikarenakan algoritma greedy tidak selalu berhasil memberikan solusi yang optimal.

Jika persoalan *travelling salesman problem* diproyeksikan ke dalam elemen-elemen algoritma *Greedy* maka keenam elemen tersebut antara lain:

1. Himpunan kandidat, C: Seluruh kota yang mungkin dikunjungi oleh pedagang
2. Himpunan solusi, S: Seluruh kota yang sudah dipilih dan terurut (berbentuk rute perjalanan) berdasarkan penentuan solusi dengan algoritma *greedy*.
3. Fungsi solusi: Memeriksa apakah rute perjalanan yang dihasilkan memiliki jarak tempuh terkecil.
4. Fungsi seleksi (*selection function*): Memilih kota yang memiliki jarak tempuh terkecil dari kota yang sedang ditempati oleh pedagang sekarang
5. Fungsi kelayakan (*feasible*): Memeriksa apakah kota yang dipilih belum pernah dikunjungi sebelumnya oleh pedagang
6. Fungsi obyektif: Meminimumkan jarak tempuh dari kota yang sedang ditempati oleh pedagang sekarang ke sebuah kota yang terhubung dengan kota tersebut.

III. Deskripsi Masalah

Mobile Legends Bang Bang (MLBB) adalah sebuah permainan daring berjenis MOBA yang dimainkan pada perangkat *smartphone* yang terkoneksi internet. Seperti permainan MOBA pada umumnya, permainan ini didasarkan kepada karakter-karakter yang digunakan untuk mengalahkan musuh dan menghancurkan menara (*turret*) yang dimiliki oleh markas musuh. Permainan ini umumnya dimainkan oleh sepuluh orang yang dibagi menjadi dua tim yaitu tim biru dan tim merah. Tim biru memiliki markas di bagian pojok kiri bawah sedangkan tim merah memiliki markan di bagian pojok kanan atas. Setiap pemain pada masing-masing tim memiliki peran yang berbeda-beda antara lain *Jungler*, *Midlaner*, *Support*, *Safelaner (Gold Lane)*, dan *Offlaner* (*Exp Lane*). *Jungler* merupakan sumber kekuatan utama pada sebuah tim sehingga harus mendapatkan level dan barang secepat mungkin untuk dapat membunuh karakter musuh. Untuk mendapatkan level dan barang dibutuhkan *gold* dan *exp* yang keduanya didapatkan dengan melakukan *farming*. *Farming* adalah sebuah mekanisme untuk mendapatkan *gold* dan *exp* sebanyak-banyaknya untuk meningkatkan level dan skil serta membeli barang yang dapat digunakan untuk mengalahkan musuh. *Farming* dilakukan dengan cara membunuh monster-monster hutan yang tersebar di arena permainan. Adapun beberapa monster hutan yang tersebar di arena permainan antara lain:

1. Lithowanderer
   Pada awal permainan, monster hutan ini akan muncul di arena permainan pada detik ke-35 dan jika dibunuh pemain akan mendapatkan 72 gold. Lithowanderer akan muncul secara acak diantara kanan dan kiri sisi sungai yang berada di tengah arena permainan.
2. Scaled Lizard
   Pada awal permainan, monster hutan ini akan muncul di arena permainan pada detik ke-40 dan jika dibunuh pemain akan mendapatkan 72 gold. Scaled Lizard akan muncul di bagian kiri atas arena permainan.
3. Serpent
   Pada awal permainan, monster hutan ini akan muncul di arena permainan pada detik ke-25 dan jika dibunuh pemain akan mendapatkan 98 gold. Serpent akan muncul di bagian kiri atas arena permainan.
4. Rockursa
   Pada awal permainan, monster hutan ini akan muncul di arena permainan pada detik ke-30 dan jika dibunuh pemain akan mendapatkan 63 gold. Rockursa akan muncul di bagian kanan bawah arena permainan.
5. Fiend
   Pada awal permainan, monster hutan ini akan muncul di arena permainan pada detik ke-25 dan jika dibunuh pemain akan mendapatkan 96 gold. Fiend akan muncul di bagian kanan bawah arena permainan.
6. Crammer
   Pada awal permainan, monster hutan ini akan muncul di arena permainan pada detik ke-40 dan jika dibunuh pemain akan mendapatkan 63 gold. Crammer akan muncul di bagian kanan bawah arena permainan.
7. Little Crab
   Pada awal permainan, monster hutan ini akan muncul di arena permainan pada detik ke-42 dan jika dibunuh pemain akan mendapatkan 58 gold. Little Crab akan muncul di bagian kiri atas dan di bagian kanan bawah.

Dari informasi diatas, dapat diketahui bahwa monster hutan yang dapat dibunuh oleh pemain yang memiliki peran *Jungler* cukup banyak, tersebar di arena permainan, dan memiliki karakteristiknya masing-masing. Setiap monster hutan memiliki waktu kemunculan yang berbeda dan jumlah perolehan gold di awal permainan yang berbeda sehingga tentunya hal ini menjadi pertimbangan pemain untuk menentukan rute *farming*. Salah satu algoritma yang dapat digunakan dalam menentukan rute *farming* adalah algoritma *Greedy*. Permasalahan ini memiliki konsep yang mirip dengan *Travelling Salesman Problem (TSP)* yang dapat diselesaikan dengan algoritma tersebut. Pemanfaatan algoritma ini bertujuan untuk membuat sebuah rute *farming* di awal permainan yang mencakup keseluruhan monster yang ada di arena permainan dengan keuntungan terbesar sehingga *farming* menjadi lebih efektif dan efisien. Adapun keuntungan dari sebuah monster hutan dihitung berdasarkan jumlah *gold* yang didapatkan dibagi dengan jarak tempuh (direpresentasikan sebagai waktu tempuh karena sulit untuk menentukan jarak di arena permainan) untuk mencapai monster hutan tersebut dari sebuah lokasi. Sebagai catatan, perolehan *gold* dari setiap monster hutan akan meningkat seiring dengan berjalannya permainan sehingga ruang lingkup persoalan ini akan diperkecil pada awal permainan saja.

Gambar 1. Penyebaran Monster Hutan MLBB
Sumber: Dokumen Pribadi dengan
Peta dari https://mobile-legends.fandom.com/wiki/Map

Keterangan:

1. Lithowanderer
2. Scaled Lizard
3. Serpent
4. Rockursa
5. Fiend
6. Crammer
7. Little Crab

IV. METODOLOGI

A. Penentuan Monster Hutan

Dari ketujuh monster hutan yang dapat dibunuh oleh pemain di awal permainan, terdapat dua monster yang memiliki spesifikasi kemunculan khusus yaitu Lithowanderer dan Little Crab. Oleh karena itu diperlukan beberapa asumsi untuk memperkecil kemungkinan solusi yang dapat dihasilkan dengan algoritma ini. Untuk Lithowanderer, penulis mengasumsikan Lithowanderer akan muncul di bagian kiri sungai sisi sungai yang berada di tengah arena permainan. Sedangkan untuk Little Crab, penulis mengasumsikan pemain akan membunuh Little Crab yang muncul di bagian kiri atas arena permainan karena Little Crab yang muncul di bagian kanan bawah berada di wilayah hutan musuh sehingga memiliki resiko tinggi untuk dibunuh karena akan memancing terjadinya pertempuran di awal permainan yang justru akan merugikan pemain jika pemain berhasil dibunuh oleh musuh. Sehingga penyebaran monster hutan yang akan dibunuh oleh pemain dapat dilihat dari gambar dibawah ini.

Gambar 2. Penyebaran Monster Hutan yang Aman dari Wilayah Musuh
Sumber: Dokumen Pribadi dengan
Peta dari https://mobile-legends.fandom.com/wiki/Map

Keterangan:

1. Lithowanderer
2. Scaled Lizard
3. Serpent
4. Rockursa
5. Fiend
6. Crammer
7. Little Crab

B. Pendataan Waktu Kemunculan dan Gold Setiap Monster Hutan

Setiap monster hutan memiliki karakteristik masing-masing, termasuk waktu kemunculan di awal permainan serta *gold* yang bisa didapatkan oleh pemain dengan membunuh monster hutan tersebut. Adapun data dari waktu kemunculan dan *gold* setiap monster hutan di awal permainan sebagai berikut:

Tabel I. Data Waktu Kemunculan dan Gold Monster Hutan

No.	Monster Hutan	Waktu Kemunculan	*Gold*
1.	Lithowanderer	35 detik	72 gold
2.	Scaled Lizard	40 detik	72 gold
3.	Serpent	25 detik	98 gold
4.	Rockursa	30 detik	63 gold
5.	Fiend	25 detik	96 gold
6.	Crammer	40 detik	63 gold
7.	Little Crab	42 detik	58 gold

C. Pendataan Waktu Tempuh Antar Monster Hutan

Setiap monster hutan yang masih hidup dapat dikunjungi dari monster hutan manapun, berbeda dengan persoalan *travelling salesman problem* yang memungkinkan adanya kota yang tidak saling terhubung. Selain itu, dikarenakan cukup sulit untuk menghitung jarak dari satu monster ke monster lainnya maka diasumsikan waktu tempuh (satuan detik) merepresentasikan jarak dari satu monster ke monster lainnya. Pencatatan waktu tempuh antar monster hutan dibuat dalam bentuk table yang merepresentasikan suatu matriks dimana setiap indeksnya merepresentasikan sebuah monster hutan. Pengindeksan tersebut didasarkan pada urutan nomor pada Tabel I. Sehingga, Lithowanderer akan memiliki indeks 1, Scaled Lizard memiliki indeks 2, begitu seterusnya hingga Little Crab yang memiliki indeks 7. Adapun data waktu tempuh antar monster hutan sebagai berikut:

Tabel II. Data Waktu Tempuh Antar Monster Hutan

	1	2	3	4	5	6	7
1	∞	6.11	4.92	11.45	12.57	15.84	5.98
2	6.11	∞	5.57	12.35	14.33	17.76	4.51
3	4.92	5.57	∞	8.00	7.84	14.75	9.88
4	11.45	12.35	8.00	∞	3.13	5.92	14.97
5	12.57	14.33	7.84	3.13	∞	5.37	16.61
6	15.84	17.76	14.75	5.92	5.37	∞	21.82
7	5.98	4.51	9.88	14.97	16.61	21.82	∞

D. Perhitungan Keuntungan Setiap Jalur Perjalanan Antar Monster

Sebagai dasar penentuan solusi menggunakan algortima *Greedy* diperlukan suatu hal yang dapat dijadikan pembanding antar simpul untuk melakukan minimasi atau maksimasi. Pada persoalan *travelling salesman problem*, hal yang dimaksud adalah jarak tempuh antar kota. Dari jarak tempuh ini akan dipilih jarak tempuh terkecil karena *travelling salesman problem* merupakan persoalan minimasi. Berbeda dengan *travelling salesman problem*, persoalan pencarian rute *farming* merupakan persoalan maksimasi sehingga dibutuhkan suatu hal berbeda untuk dijadikan pembanding antar simpulnya.

Pada persoalan ini penulis mengajukan sebuah hal yang dapat dijadikan pembanding antar simpul yaitu keuntungan. Keuntungan didefinisikan sebagai jumlah *gold* yang didapatkan setelah membunuh monster hutan tersebut dibagi waktu kemunculan (untuk penentuan simpul awal) atau dibagi waktu tempuh antar monster hutan (untuk penentuan simpul setelah simpul awal). Adapun rumus untuk perhitungan keuntungan dan tabel keuntungan setiap monster hutan pada simpul awal sebagai berikut:

$$P(i) = \frac{G(i)}{t(i)}$$

Keterangan:

1. P(i) = Keuntungan monster hutan ke-i
2. G(i) = *Gold* yang didapatkan jika pemain membunuh monster ke-i
3. t(i) = Waktu kemunculan monster hutan ke-i diawal permainan

Tabel III. Data Keuntungan Monster Hutan Pada Simpul Awal

No.	Monster Hutan	G(i)	t(i)	P(i)
1.	Lithowanderer	72 gold	35 detik	2.06
2.	Scaled Lizard	72 gold	40 detik	1.8
3.	Serpent	98 gold	25 detik	3.92
4.	Rockursa	63 gold	30 detik	2.1
5.	Fiend	96 gold	25 detik	3.84
6.	Crammer	63 gold	40 detik	1.58
7.	Little Crab	58 gold	42 detik	1.38

Sedangkan rumus untuk perhitungan keuntungan dan tabel keuntungan setiap monster hutan pada simpul setelah simpul awal sebagai berikut:

$$P(i,j) = \frac{G(j)}{t(i,j)}$$

Keterangan:
1. P(i,j) = Keuntungan monster hutan ke-j jika pemain bergerak dari monster hutan ke-i
2. G(j) = *Gold* yang didapatkan jika pemain membunuh monster ke-j
3. t(i, j) = Waktu tempuh dari monster hutan ke-i menuju monster hutan ke-j

Tabel IV. Data Keuntungan Monster Hutan Pada Simpul Setelah Simpul Awal

	1	2	3	4	5	6	7
1	∞	11.78	19.91	5.50	7.63	3.97	9.69
2	11.78	∞	17.59	5.10	6.70	3.54	12.86
3	14.63	12.92	∞	7.88	12.24	4.27	5.47
4	6.28	5.82	12.25	∞	30.67	10.64	3.87
5	5.72	5.02	12.50	20.13	∞	11.73	3.49
6	4.54	4.05	6.64	10.64	17.87	∞	2.66
7	12.04	15.96	9.91	4.20	5.78	2.89	∞

E. Penerapan Algoritma Greedy

Untuk menyelesaikan persoalan rute *farming* ini harus diawali dengan penentuan setiap elemen *Greedy* pada persoalan ini. Adapun proyeksi dari keenam elemen *Greedy* pada persoalan ini antara lain:
1. Himpunan kandidat, C: Seluruh monster hutan yang dapat dibunuh oleh pemain
2. Himpunan solusi, S: Seluruh monster hutan yang sudah dipilih dan terurut (berbentuk rute perjalanan) berdasarkan penentuan solusi dengan algoritma *greedy*.
3. Fungsi solusi: Memeriksa apakah rute perjalanan yang dihasilkan memiliki keuntungan terbesar
4. Fungsi seleksi (*selection function*): Memilih monster hutan yang memiliki keuntungan terbesar dari monster hutan yang sedang dibunuh oleh pemain saat ini
5. Fungsi kelayakan (*feasible*): Memeriksa apakah monster hutan yang dipilih belum pernah dibunuh sebelumnya oleh pemain
6. Fungsi obyektif: Memaksimalkan keuntungan dari monster hutan yang sedang dibunuh oleh pemain saat ini ke sebuah monster hutan yang belum dibunuh oleh pemain

Setelah itu dapat ditentukan simpul awal sebagai monster hutan yang pemain harus bunuh pertama kali yang memiliki keuntungan terbesar. Berdasarkan Tabel III, didapatkan bahwa simpul awal adalah monster hutan ke-3 (Serpent) dengan total keuntungan 3.92. Berikutnya akan diselesaikan dengan algoritma *Greedy* dengan langkah-langkah sebagai berikut:

1. Iterasi 1:
   Simpul yang dipilih: 3 (Simpul awal dengan keuntungan terbesar)
   Rute: 3
   Total Keuntungan: 3.92
2. Iterasi 2:
   Data keuntungan simpul lain yang dapat dipilih dari simpul 3: 14.63, 12.92, 7.88, 12.24, 4.27, 5.47.
   Keuntungan maksimum: 14.63
   Simpul yang dipilih: 1 (Simpul selanjutnya dengan keuntungan maksimum)
   Rute: 3 - 1
   Total Keuntungan: 18.55
3. Iterasi 3:
   Data keuntungan simpul lain yang dapat dipilih dari simpul 1: 11.78, 5.50, 7.63, 3.97, 9.69.
   Keuntungan maksimum: 11.78
   Simpul yang dipilih: 2 (Simpul selanjutnya dengan keuntungan maksimum)
   Rute: 3 - 1 - 2
   Total Keuntungan: 30.33
4. Iterasi 4:
   Data keuntungan simpul lain yang dapat dipilih dari simpul 2: 5.10, 6.70, 3.54, 12.86.
   Keuntungan maksimum: 12.86
   Simpul yang dipilih: 7 (Simpul selanjutnya dengan keuntungan maksimum)
   Rute: 3 - 1 - 2 - 7
   Total Keuntungan: 43.19
5. Iterasi 5:
   Data keuntungan simpul lain yang dapat dipilih dari simpul 7: 4.20, 5.78, 2.89.
   Keuntungan maksimum: 5.78
   Simpul yang dipilih: 5 (Simpul selanjutnya dengan keuntungan maksimum)
   Rute: 3 - 1 - 2 - 7 - 5
   Total Keuntungan: 48.97
6. Iterasi 6:
   Data keuntungan simpul lain yang dapat dipilih dari simpul 5: 20.13, 11.73.
   Keuntungan maksimum: 20.13
   Simpul yang dipilih: 4 (Simpul selanjutnya dengan keuntungan maksimum)
   Rute: 3 - 1 - 2 - 7 - 5 - 4
   Total Keuntungan: 69.10

7. Iterasi 7:
   Keuntungan maksimum: 10.64
   Simpul yang dipilih:6 (Simpul yang tersisa)
   Rute: 3 - 1 - 2 - 7 - 5 - 4 - 6
   Total Keuntungan: 79.74

Dengan demikian, rute *farming* di awal permainan yang mencakup keseluruhan monster yang ada di arena permainan dengan keuntungan terbesar yang ditelusuri dengan algoritma *Greedy* adalah 3 → 1 → 2 → 7 → 5 → 4 → 6 (Serpent → Lithowanderer → Scaled Lizard → Little Crab → Fiend → Rockursa → Crammer) dengan total keuntungan sebesar 79.74.

Gambar 3. Hasil Pencarian Rute *Farming* dengan Algoritma *Greedy*
Sumber: Dokumen Pribadi dengan
Peta dari https://mobile-legends.fandom.com/wiki/Map

Persoalan ini juga dapat diselesaikan dengan menggunakan sebuah program. Program yang digunakan untuk menyelesaikan persoalan ini menggunakan algoritma *Greedy* yang telah dijelaskan sebelumnya dan dibuat dengan bahasa pemrograman Python serta paradigma pemrograman prosedural. Adapaun *pseudocode* dari program tersebut sebagai berikut:

```
procedure farmingRoute()
{Program akan menampilkan rute farming di awal permainan yang mencakup keseluruhan monster yang ada di arena permainan dengan keuntungan terbesar yang ditelusuri dengan algoritma Greedy}

Kamus Lokal
i, next: integer
arrGold, arrSpawn, currRoute: array of integer
arrProfit: array of array of integer

function maximumStart(input arrGold: array of integer, input arrSpawn: array of integer, input n : integer)→ res : integer
{Fungsi ini digunakan untuk mendapatkan simpul awal yang memiliki keuntungan terbesar}

function maximumNext(input arrProfit: array of array of integer, input currRoute: array of integer, input n : integer)→ res : integer
{Fungsi ini digunakan untuk mendapatkan selanjutnya yang memiliki keuntungan terbesar dari simpul terakhir yang dikunjungi}

function len(input arr: array of integer) → length : integer
{Fungsi ini akan mengembalikan panjang dari array of integer}

procedure append(input/output arr: array of integer, input x : integer)
{Fungsi ini akan menambahkan x ke arr}

Algoritma
{Deklarasi data yang diperlukan}
arrGold ← [72, 72, 98, 63, 96, 63, 58]
arrSpawn ← [35, 40, 25, 30, 25, 40, 42]
arrProfit ← [[-1, 11.78, 19.91, 5.50, 7.63, 3.97, 9.69], [11.78, -1, 17.59, 5.10, 6.70, 3.54, 12.86], [14.63, 12.92, -1, 7.88, 12.24, 4.27, 5.47], [6.28, 5.82, 12.25, -1, 30.67, 10.64, 3.87], [5.72, 5.02, 12.50, 20.13, -1, 11.73, 3.49], [4.54, 4.05, 6.64, 10.64, 17.87, -1, 2.66], [12.04, 15.96, 9.91, 4.20, 5.78, 2.89, -1],]

{Dapatkan simpul awal yang maksimum}
currRoute ← []
append(currRoute, maximumStart(arrGold, arrSpawn, 7)
while length(currRoute) < 7 do
     next ← maximumNext(arrProfite, currRoute, 7)
     append(currRoute, next)

{Tampilkan Hasilnya}
output(currRoute)
```

Dari program tersebut dilakukan eksperimen dengan data yang sesuai dengan Tabel III dan Tabel IV. Hasil dari eksperimen tersebut sesuai dengan penelusuran solusi dengan algoritma *Greedy* sebelumnya yaitu 3 → 1 → 2 → 7 → 5 → 4 → 6. Sehingga solusi ini dapat disimpulkan valid.

```
D:\Institut Teknologi Bandung\IF\Semester 4\Stima\Makalah>py farmingRoute.py
3 - 1 - 2 - 7 - 5 - 4 - 6
```

Gambar 4. Hasil Eksperimen Pencarian Rute *Farming* dengan Program
Sumber: Dokumen Pribadi

V. KESIMPULAN

Popularitas permainan daring semakin meningkat seiring perkembangan teknologi. Hal ini terbukti dengan perkembangan pesat dari salah satu jenis permainan daring yaitu *Multiplayer Online Battle Arena (MOBA)* khususnya permainan Mobile Legends Bang Bang (MLBB). Hal ini dikarenakan mobilitas dari permainan daring tersebut yang cukup baik (bisa dimainkan di *smartphone* yang terkoneksi dengan internet) serta tingkat kesulitan permainan yang tidak terlalu tinggi. Seperti permainan berjeni MOBA pada umumnya, di permainan Mobile Legends Bang Bang (MLBB) dikenal sebuah istilah yaitu *farming*. *Farming* merupakan sebuah mekanisme untuk mendapatkan *gold* dan *exp* sebanyak-banyaknya untuk meningkatkan level

dan skil serta membeli barang yang dapat digunakan untuk mengalahkan musuh dengan cara membunuh monster hutan yang tersebar di arena permainan. Oleh karena itu, diperlukan sebuah rute *farming* di awal permainan yang mencakup keseluruhan monster yang ada di arena permainan dengan keuntungan terbesar sehingga *farming* menjadi lebih efektif dan efisien khususnya untuk pemain yang memiliki peran *Jungler*.

Dengan Algoritma *Greedy*, dapat dihasilkan sebuah rute *farming* di awal permainan yang memiliki keuntungan terbesar yang dapat digunakan oleh pemain khususnya yang memiliki peran *Jungler*. Adapun rute *farming* tersebut adalah Serpent → Lithowanderer → Scaled Lizard → Little Crab → Fiend → Rockursa → Crammer dengan total keuntungan sebesar 79.74. Dengan rute *farming* tersebut diharapkan pemain dapat lebih efektif dalam melakukan *farming* sehingga pemain dapat lebih cepat dalam mendapatkan *gold* dan *exp*. Tentunya hal ini sangat penting mengingat *Jungler* merupakan sumber kekuatan utama sebuah tim. Akan tetapi, tidak selamanya rute tersebut menjadi rute terbaik karena simpul yang diambil dalam setiap iterasinya merupakan simpul dengan keuntungan terbesar yang dapat dipilih pada iterasi tersebut (optimum lokal). Sehingga dapat disimpulkan bahwa sebenarnya rute ini hanyalah aproksimasi dari rute *farming* yang paling efektif. Untuk mendapatkan rute *farming* yang lebih efektif diperlukan eksperimen lebih lanjut dengan algoritma lainnya seperti *Branch and Bound, A*, Dynamic Programming*, ataupun algoritma lainnya yang dapat menyelesaikan persoalan optimasi pencarian rute dan sebagai catatan bahwa rute *farming* akan lebih efektif jika keadaannya sesuai.

VI. Ucapan Terima Kasih

Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa karena atas berkat dan rahmat-Nya yang melimpah, penulis dapat menyelesaikan makalah ini. Penulis juga ingin mengucapkan terima kasih kepada Bapak Ir. Rila Mandala, M.Eng.,Ph.D. selaku dosen mata kuliah IF2211 Strategi Algoritma Kelas K1 serta Bapak Dr. Ir. Rinaldi Munir, MT. yang telah menyediakan bahan ajar melalui situs web sehingga memudahkan penulis dalam memahami mata kuliah ini. Penulis juga mengucapkan terima kasih kepada kedua orang tua dan keluarga penulis yang senantiasa mendoakan dan mendukung studi penulis.

Pranala Video Youtube

Untuk memberikan gambaran yang lebih jelas tentang makalah ini, telah dibuat video penjelasan yang dapat diakses melalui tautan bit.ly/VideoMakalah13519026

Referensi

[1] Wiyanti, D. T. (2013). Algoritma Optimasi Untuk Penyelesaian Travelling Salesman Problem. Jurnal Transformatika, 11(1), 1-6.

[2] Ahdiyat, M. A. (2018). Analisis Keterlibatan Komunitas dalam Industri Permainan Daring di Indonesia. Interaksi: Jurnal Ilmu Komunikasi, 7(2), 105-115.

[3] Munir, Rinaldi. (2021, Januari 27). *Greedy Bagian 1,2, dan 3*. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/20202021/stima20-21.htm, diakses pada 11 Mei 2021.

[4] Mobile Legends Live Player Count and Statistics. https://activeplayer.io/mobile-legends-bang-bang-live-player-count-and-statistics/, diakses pada 11 Mei 2021.

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, 11 Mei 2021

Gde Anantha Priharsena / 13519026