Implementasi Algoritma Greedy dalam Menentukan Pola Farming Berdasarkan Keputusan Ganking Core pada Permainan Mobile Legends

Rolland Steven Supardi 13519173
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519173@std.stei.itb.ac.id

***Abstract*—Algoritma sering digunakan dalam ilmu komputer. Hal tersebut berupa rangkaian instruksi yang penting digunakan untuk memecahkan beragam persoalan dari yang sederhana hingga kompleks. Salah satunya berupa pengembangan permainan dan algoritma yang cukup populer digunakan adalah algoritma *greedy*. Algoritma *greedy* dapat digunakan dalam permainan untuk meningkatkan peluang kemenagan dengan memaksimalkan perolehan atribut yang disediakan dalam permainan. Salah satu atribut permainan yang cukup populer terutama dalam permainan strategi adalah *gold*. Algoritma *greedy* dapat digunakan dalam rangka memaksimalkan pendapatan *gold* dalam permainan Mobile Legends: Bang Bang.**

***Keywords*—algoritma, *greedy*, *gold*, maksimum**

I. PENDAHULUAN

Algoritma merupakan serangkaian instruksi atau langkah-langkah terstrukstur dan sistematis yang digunakan untuk menyelesaikan suatu persoalan. Algoritma memegang peranan yang penting dalam ilmu komputer untuk menyelesaikan persoalan secara efektif dan efisien. Salah satu algoritma yang cukup populer digunakan adalah algoritma *greedy*. Algoritma *greedy* merupakan algoritma yang digunakan untuk permasalahan optimasi. Oleh sebab itu, algoritma ini cukup populer pada permainan strategi terutama dalam rangka memaksimumkan atribut permainan untuk mencapai kemenagan.

Pada permainan strategi, terutama permainan *action real-time strategy* seperti Mobile Legends: Bang Bang, perolehan

Gambar 1. Logo Mobile Legends: Bang Bang
(Sumber:https://www.facebook.com/MobileLegendsGameIndonesia/photos/a.820752248076591/2250322665119535/)

gold menjadi salah satu faktor penentu kemenangan. Kegiatan mencari atau memperoleh *gold* dalam permainan dikenal dengan istilah *farming*. *Farming* merupakan istilah yang cukup populer pada permainan strategi seperti Mobile Legends. Mobile Legends: Bang Bang atau yang biasa dikenal sebagai Mobile Legends merupakan permainan yang cukup populer sejak tahun 2016. Permainan ini merupakan permainan *action real-time strategy* dengan tujuan utama menghancurkan menara lawan sambil mempertahankan menara sendiri.

Seperti yang sudah disebutkan sebelumnya, *farming* merupakan kegiatan yang penting dalam memperoleh kemenangan. Tujuan utama dari *farming* adalah memperoleh *gold* sebanyak mungkin hingga dapat mengungguli *gold* lawan. *Gold* dalam permainan Mobile Legends dapat diperoleh dari membunuh monster hutan. Setiap monster hutan di dalam peta permainan terletak pada posisi tertentu dengan waktu *spawn* dan perolehan *gold* yang berbeda satu sama lain. Dalam menghadapi hal ini, algoritma *greedy* dapat digunakan dalam menentukan pola *farming* berdasarkan monster hutan yang tersedia di sekitar pemain dan *gold* yang diperoleh jika membunuhnya. Hal ini akan meningkatkan persentase kemenangan dengan harapan memperoleh *gold* yang paling banyak pada setiap langkah *farming* yang diambil. Pada makalah ini, penulis akan membahas mengenai implementasi algoritma *greedy* untuk menentukan pola *farming* sehingga diperoleh *gold* maksimum pada setiap langkahnya.

II. LANDASAN TEORI

A. Algoritma Greedy

Algoritma *greedy* merupakan algoritma yang cukup populer dalam menyelesaikan persoalan optimasi. Hal ini disebabkan algoritma ini cukup sederhana dengan memilih pilihan terbaik yang dapat diambil dalam suatu waktu tanpa memikirkan dampak ke depannya. Persoalan optimasi yang erat kaitannya dengan algoritma ini merupakan persoalan yang mencari solusi optimal. Oleh karena itu, solusi optimal yang dicari dapat berupa solusi yang maksimum maupun minimum. Persoalan optimasi yang mencari solusi maksimum disebut

sebagai persoalan maksimasi (*maximization).* Salah satu contoh dari persoalan maksimasi adalah persoalan *knapsack*. Persoalan ini bertujuan mendapatkan barang dengan nilai barang yang paling tinggi. Sedangkan jenis persoalan dengan tujuan mencari solusi minimum disebut sebagai persoalan minimasi (*minimization*). Salah satu contoh dari persoalan ini ialah persoalan penukaran uang. Persoalan ini bertujuan menukarkan uang dengan jumlah koin sedikit mungkin.

Algoritma *greedy* seperti namanya merupakan algoritma yang rakus. Algoritma ini terkenal dengan prinsipnya yaitu mengambil segala sesuatu yang dapat diambil saat ini (hal yang memberikan keuntungan saat itu). Algoritma *greedy* digunakan untuk menyelesaikan permasalahan optimasi secara langkah demi langkah. Seperti yang sudah disebutkan di atas algoritma ini akan mengambil langkah terbaik tanpa memperhatikan konsekuensi ke depannya (*take what you can get now)* sambil berharap bahwa langkah yang diambil akan mengarah pada optimum global. Oleh karena itu, setiap langkah yang diambil dalam algoritma *greedy* bersifat optimum lokal saja, belum tentu optimum global.

Berdasarkan paparan di atas, sangat jelas terlihat bahwa algoritma *greedy* bersifat tidak optimal sebab algoritma ini tidak memberikan jaminan bahwa solusi yang diperoleh merupakan solusi yang optimal. Seperti yang diketahui bahwa apa yang terlihat baik saat ini belum tentu akan mendatangkan kebaikan di akhirnya. Selain itu, algoritma *greedy* tidak melakukan pencarian secara menyeluruh terhadap seluruh kemungkinan solusi, kedua hal tersebut menjadi titik lemah dari algoritma *greedy.*

Secara umum terdapat enam elemen utama dalam algoritma *greedy*, di antaranya:

1. Himpunan kandidat (C)

   Himpunan dari elemen-elemen yang akan dipilih sebagai solusi

2. Himpunan solusi (S)

   Himpunan dari elemen yang sudah terpilih sebagai solusi

3. Fungsi solusi

   Fungsi untuk menentukan apakah himpunan solusi sudah memberikan solusi

4. Fungsi seleksi

   Fungsi yang digunakan dalam memilih elemen sebagai solusi secara *greedy*

5. Fungsi kelayakan

   Fungsi untuk memeriksa elemen yang dipilih melanggar batasan atau tidak agar layak sebagai suatu solusi

6. Fungsi obyektif

   Fungsi untuk memaksimumkan atau meminimumkan solusi

Berdasarkan keenam elemen tersebut, algoritma *greedy* akan melakukan pencarian dengan mengisi himpunan bagian S dari himpunan kandidat C. Dalam hal tersebut, digunakan beberapa fungsi untuk memastikan solusi merupakan solusi yang layak sehingga didapatkan S yang menyatakan solusi. Pada akhirnya S akan dimaksimumkan atau diminimumkan dengan fungsi obyektif.

B. Permainan Mobile Legends: Bang Bang

Mobile Legends: Bang Bang merupakan permainan *mobile action real-time strategy* yang sangat populer hingga saat ini. Permainan yang dirilis dan dikembangkan oleh perusahaan bernama Moonton ini dimainkan secara daring dengan melibatkan sepuluh pemain dalam satu pertandingan. Mobile Legends: Bang Bang pertama kali dirilis untuk *platform* Android pada tanggal 14 Juli 2016. Barulah kemudian pada tanggal 9 November 2016, permainan ini dapat dimainkan di *platform* iOS.

Gambar 2. *Loading screen* Mobile Legends

(Sumber: https://img.esportsku.com/wp-content/uploads//2020/04/Mobile-Legends-Update-Patch-note-1.4.70.jpeg)

Mobile Legends: Bang Bang merupakan permainan strategi 5v5 (lima lawan lima) yang berlatar belakang kerajaan. Sebutan bagi latar tempat berdirinya kerajaan-kerajaan tersebut adalah Land of Dawn. Tempat ini merupakan peta yang digunakan dalam permainan di mana nantinya para pemain berkompetisi dengan strategi yang sudah diterapkan dalam tim masing-masing. Tujuan utama dari permainan strategi ini adalah menghancurkan menara tim lawan dengan memanfaatkan sumber daya yang ada pada peta. Secara umum pemain yang merupakan peran inti harus melakukan *farming* untuk memperoleh *gold* yang banyak hingga mengungguli lawan dan membantu rekan tim lainnya untuk menghancurkan menara. Dalam hal ini, terdapat beberapa istilah penting yang perlu diperhatikan dalam permainan untuk meraih kemenangan, di antaranya:

1. *Farming*

   Farming merupakan kegiatan yang dilakukan untuk memperoleh *gold* sebanyak-banyaknya dalam permainan. Kegiatan tersebut dilakukan dengan cara membunuh monster hutan dalam rangka meningkatkan kekuatan pemain dan membantu tim untuk menghancurkan menara lawan dengan mudah.

2. *Buff*

Buff merupakan atribut tambahan yang diperoleh pemain ketika membunuh monster hutan tertentu. Pada permainan Mobile Legends: Bang Bang terdapat lima monster hutan yang memberikan efek *buff* ini, di antaranya Serpent, Fiend, Turtle, Crab, dan Lithowanderer.

3. *Ganking*

   Ganking merupakan salah satu strategi yang sering digunakan dalam permainan berbasis strategi *real time*. Strategi ini secara umum dilakukan dengan melakukan penyergapan musuh yang sedang berada di luar posisi menara atau jangkauan bantuan temannya.

4. Lord

   Lord merupakan monster yang muncul pada menit kesembilan permainan. Monster ini dapat memberikan bantuan untuk menghancurkan menara lawan ketika pemain berhasil membunuhnya, Lord akan bertindak sebagai pemain keenam dalam permainan yang siap untuk menghancurkan menara lawan.

5. *Roaming*

   Roaming merupakan istilah yang mengacu pada perpindahan jalur dalam peta. Dalam satu peta terdapat tiga jalur dan masing-masing jalur terdiri atas tiga menara luar yang saling terhubung kepada menara utama sebagai target yang ingin dihancurkan. Strategi ini dilakukan untuk membantu teman yang sedang dalam bahaya terkena *ganking* atau melakukan *ganking* kepada lawan.

Secara singkat, alur permainan hingga mencapai kemenangan dalam permainan Mobile Legends adalah sebagai berikut:

1. Kelima pemain terjun dalam peta permainan dan mengisi ketiga jalur untuk menjaga menara dari serangan lawan.
2. Pemain dengan peran inti langsung melakukan *farming* untuk mengungguli *gold* lawan dan melakukan *roaming* jika dirasa memiliki peluang keberhasilan *ganking* yang tinggi.
3. Ketika berhasil melakukan *ganking*, menara lawan bebas dari penjagaan untuk sementara, pemain inti beserta pemain penjaga menara pada suatu jalur mulai melakukan penyerangan terhadap menara hingga tim lawan mulai tiba untuk menjaga menara kembali.
4. Ulangi langkah satu hingga tiga sampai akhirnya semua menara luar berhasil dihancurkan.
5. Apabila pemain merasa kesulitan untuk menghancurkan menara utama karena kelima musuh terpaksa berkumpul untuk menjaga menara utama, pemain dapat menyusun strategi untuk mendapatkan Lord.
6. Bersama dengan Lord, kelima pemain berkumpul dan fokus untuk menghancurkan menara utama musuh.
7. Menara utama berhasil dihancurkan dan permainan berakhir.

Secara umum dalam satu tim yang bertanding, pemain dapat dibedakan menjadi empat peran:

1. *Tanker*

   Tanker merupakan peran bagi pemain yang berada di jalur tengah. Pemain ini memiliki peran untuk melindungi pemain inti dari *ganking* musuh dan melakukan inisiasi terhadap lawan jika ingin melakukan *ganking*.

2. *Support*

   Support merupakan sebutan bagi pemain yang bertugas menjaga menara di jalur tengah. Seperti namanya, selain menjaga menara di jalur tengah, seorang *Support* membantu pemain lainnya ketika berperang ataupun melarikan diri dari *ganking*.

3. *Core*

   Core merupakan sebutan bagi pemain inti. Pemain ini bertugas untuk melakukan *farming*, meraih *gold* yang lebih unggul dari lawan, dan membantu pemain lain untuk menghancurkan menara di jalurnya.

4. *Offlaner*

   Offlaner merupakan sebutan bagi pemain yang bertugas menjaga menara di jalur samping. Berbeda dengan *Support* yang didampingi oleh *Tanker* dan *Core* di jalur tengah, seorang *Offlaner* harus menjaga menara seorang diri, berhati-hati terkena *ganking* dari musuh, dan mencuri kesempatan ketika lawan sedang meninggalkan menara atau membantu *Core* ketika akan melakukan *ganking*.

Gambar 3. Peta Mobile Legends beserta ketiga jalurnya

(Sumber : https://assets.pikiran-rakyat.com/crop/0x0:0x0/750x500/photo/2020/09/28/1329319574.jpg)

Berkaitan dengan peran di atas, dapat dilihat bahwa *Core* memegang peranan yang sangat penting. Pemain ini dipercayakan oleh tim untuk menguasai sumber daya yang tersedia di peta sehingga mendapatkan *gold* yang lebih unggul dan membantu pemain dalam melakukan *ganking* terhadap lawan di suatu jalur untuk menghancurkan menara. Dalam melakukan *farming* dan *ganking* secara efektif, seorang *Core* harus dapat memperoleh jumlah *gold* sebanyak mungkin dalam

perjalanannya hingga mencapai suatu jalur yang ingin dibantu untuk melakukan *ganking*. Oleh karena itu, penulis merasa pola *farming* seperti ini cocok diimplementasikan dengan algoritma *greedy*. Seorang *Core* memiliki tiga pilihan tujuan akhir dari simpul awalnya (menara utama tim) yakni melakukan *ganking* pada jalur atas, tengah, atau bawah. Selama perjalanan menuju tujuan akhir, seorang *Core* dihadapkan pada beberapa persimpangan yang di setiap daerah tersebut terdapat monster hutan, penulis merasa algoritma *greedy* cocok diterapkan untuk memaksimalkan perolehan *gold* seorang *Core* dalam perjalanan melakukan *ganking* pada salah satu jalur.

C. *Gold*

Gold merupakan salah satu atribut pada permainan yang dimiliki oleh setiap pemain dalam pertandingan. *Gold* bertindak layaknya mata uang dalam pertandingan yang dapat digunakan untuk memperkuat pemain dalam pertandingan. Atribut ini dapat diperoleh dengan berbagai cara, salah satunya dengan melakukan *farming*, membunuh *minions*, menghancurkan menara, membunuh Turtle, membunuh Lord, atau membunuh pemain lain. Dalam makalah ini, penulis akan menekankan perolehan *gold* dalam mekanisme *farming*, berikut monster hutan dan jumlah *gold* yang didapatkan ketika membunuhnya:

1. Fiend : 75 *gold*
2. Crammer: 81 *gold*
3. Rockursa: 63 *gold*
4. Lithowanderer: 72 *gold*
5. Crab : 58 *gold*
6. Serpent : 104 *gold*
7. Scaled Lizard : 82 *gold*

III. Hasil dan Pembahasan

Algoritma *greedy* merupakan algoritma yang rakus, mengambil yang terbaik saat ini tanpa memperhitungkan hasil akhirnya nanti. Algoritma ini digunakan pada persoalan optimasi baik untuk meminimalkan solusi maupun memaksimalkan solusi. Pada permainan Mobile Legends: Bang Bang, algoritma ini dapat digunakan untuk membantu pola *farming* seorang *Core* terhadap keputusan *ganking* yang diambil sehingga didapatkan *gold* terbanyak hingga tiba di suatu jalur. Secara sederhana, seorang *Core* akan dihadapkan pada pilihan monster hutan yang akan dibunuh dalam perjalanan melakukan *ganking* terhadap suatu jalur dari menara utama. Dengan algoritma *greedy*, *Core* akan membunuh monster yang akan memberikan jumlah *gold* terbesar saat itu, terus berlanjut hingga tiba di salah satu jalur untuk melakukan *ganking*.

Penulis telah membuat simulasi dari posisi monster hutan pada peta permainan secara sederhana terkait pola *farming* dalam perjalanan *ganking* dengan algoritma *greedy*, implementasinya dilakukan dalam bahasa Python.

A. Tahap Persiapan Simulasi

Berikut adalah penjabaran dari enam elemen utama algoritma *greedy* pada simulasi penentuan pola *farming* dalam perjalanan *ganking* seorang *Core:*

1. Himpunan kandidat (C)

   Himpunan dari monster-monster hutan yang dapat dibunuh untuk perolehan *gold.*

2. Himpunan solusi (S)

   Himpunan dari monster hutan yang sudah terpilih dan akan dibunuh untuk memperoleh *gold.*

3. Fungsi solusi

   Fungsi untuk menentukan apakah himpunan solusi sudah memberikan solusi, yaitu apakah *Core* sudah mencapai simpul tujuan untuk melakukan *ganking.*

4. Fungsi seleksi

   Fungsi yang digunakan dalam memilih elemen sebagai solusi secara *greedy*, yaitu memilih monster hutan yang menghasilkan *gold* paling banyak pada setiap persimpangan.

5. Fungsi kelayakan

   Fungsi untuk memeriksa monster yang dibunuh berada pada jalur *ganking* yang ditentukan *Core*.

6. Fungsi obyektif

   Fungsi untuk memaksimumkan perolehan *gold* dengan membunuh monster hutan yang menghasilkan *gold* terbanyak pada setiap pilihan jalurnya.

Berikut algoritma *greedy* yang diterapkan pada simulasi tersebut:

1. *Core* menentukan simpul tujuan *ganking*, simpul tujuan berupa jalur atas, tengah, atau bawah.
2. Setelah itu, *Core* mulai memilih monster hutan dengan perolehan *gold* terbesar.
3. Periksa apakah monster hutan tersebut berada pada jalur *ganking* yang dipilih.
4. Jika iya, maka *Core* akan membunuh monster tersebut.
5. Jika tidak, maka *Core* akan memilih monster dengan perolehan *gold* terbesar lainnya yang ada dalam pilihan saat itu.
6. Ulangi langkah dua sampai lima hingga *Core* mencapai simpul tujuan

Berikut *pseudocode* dari fungsi *farming* pada simulasi yang menerapkan algoritma *greedy* :

```
function farming(input g: graph, tujuan: string)→ array of string


Kamus Data:
  hasil: array of string
  i, key, current: string
  gold, current_neighbors: Map {berisi monster sebgai key dan value gold yang
```

```
dihasilkan}
  jalur: Map {key simpul dan value posisi jalur monster(Atas/Tengah/Bawah}
  layak: boolean {true jika monster berada pada jalur ganking yang dipilih}
  max: integer
Algoritma:
  hasil ← ["Core"] {inisialisasi dengan simpul awal}
  current ← "Core" {tinjau dari simpul awal}
  while (current≠tujuan) do {memeriksa apakah sudah mencapai solusi}
    current_neighbors ← {} {menampung simpul hidup, yaitu simpul tetangga dari simpul yang sedang ditinjau}
    layak ← False
    max ← 0 {menampung nilai gold terbesar}
    {membangkitkan simpul tetangga}
    for i in (list simpul tetangga dari current) do
      if(i not in hasil) then
        current_neighbors_i←gold_i
        {memasukkan nama simpul tetangga beserta gold yang dihasilkan}
    {melakukan seleksi simpul yang memberikan gold terbesar}
    for key in current_neighbors do
      if(current_neighbors_key > max) then
        for i in jalur_key do
        {periksa kelayakan, jika simpul berada di jalur ganking Core}
          if (i = tujuan) then
            layak ← True
        if (layak) then
          max ← current_neighbors_key
          current ← key {meninjau simpul selanjutnya dengan gold terbesar}
    {memasukkan monster dengan gold terbesar ke dalam hasil}
  → hasil
```

Dapat dilihat pada fungsi di atas terjadi pengulangan *while* hingga mencapai simpul solusi dan terus dilakukan pembaruan terhadap *gold* yang dihasilkan tiap simpul dari awal sampai tujuan. Berdasarkan komputasi yang dilakukan di dalam fungsi, kompleksitas waktu dari algoritma *greedy* yang diterapkan pada fungsi *farming* adalah $O(n^2)$.

B. Tahap Pelaksanaan Simulasi

Dalam program simulasi yang telah dibuat terdapat beberapa variabel yang digunakan antara lain:

1. hasil: variabel yang menampung elemen monster hutan yang terpilih sebagai solusi persoalan
2. i, *key*, dan *current*: variabel yang menampung nama monster baik sebagai *iterator* maupun variabel sementara
3. *gold*: variabel peta yang menampung seluruh monster hutan sebagai *key*nya dan *gold* yang dihasilkan sebagai *value*nya
4. *current_neighbors*: variabel *map* yang menampung monster hutan tetangga sebagai *key*nya dan *gold* yang dihasilkan sebagai *value*nya
5. jalur: variabel *map* yang menampung seluruh monster hutan sebagai *key*nya dan jalur *ganking* monster sebagai *value*nya
6. layak: variabel *boolean* yang bernilai *true* jika monster berada pada jalur *ganking* yang sesuai dengan yang dikehendaki *Core*.
7. *max*: variabel yang menampung nilai *gold* terbesar saat itu.

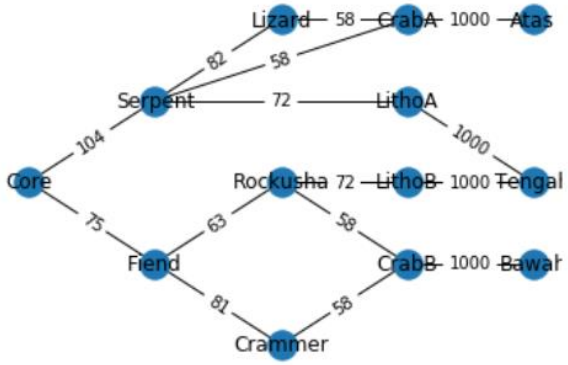
Berikut alur berjalannya program simulasi pola *farming* berdasarkan keputusan jalur *ganking* Core:

1. Dibuat sebuah graf yang berisi monster hutan sebagai simpulnya dan tentukan ketetangaannya atau keterhubungannya satu sama lain.
2. Setelah itu, dibuat empat struktur data *map* yaitu *pos*(menampung posisi x dan y monster terhadap peta permainan/hanya untuk keperluan visualisasi), *jalur*(menampung jalur *ganking* monster berada), *weight* (menampung bobot sisi berdasarkan *gold* yang dihasilkan monster tujuan/hanya untuk keperluan visualisasi), *gold (*menampung *gold* yang dihasilkan monster hutan).
3. Melakukan visualisasi graf peta permainan dengan monster hutan sebagai simpulnya dan bobot sisinya berdasarkan *map weight*.
4. Ditentukan graf dan simpul *tujuan* yang ingin ditelusuri.
5. Menginisialisasi variabel *hasil* dan *current* dengan simpul awal.
6. Melakukan pengulangan terhadap simpul dengan memilih monster hutan yang sesuai dengan fungsi solusi, fungsi seleksi, dan fungsi kelayakan.

7. Pada setiap langkahnya/perulangan, periksa monster hutan kemudian pilih yang menghasilkan *gold* terbesar dan berada di jalur *ganking Core*.
8. Memasukkan monster hutan yang sesuai ke variabel *hasil* dan *current* untuk ditinjau lebih lanjut.
9. Ulangi langkah enam hingga delapan sampai mencapai simpul *tujuan*.

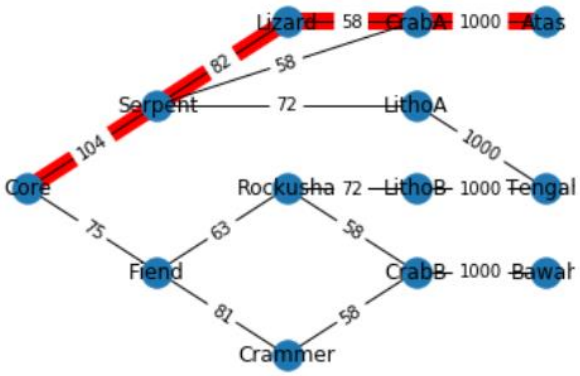
C. Tahap Pengujian Simulasi

Berikut hasil tangkapan layar dari program simulasi pola *farming* berdasarkan keputusan jalur *ganking Core:*



Gambar 4. Visualisasi peta Mobile Legends pada simulasi

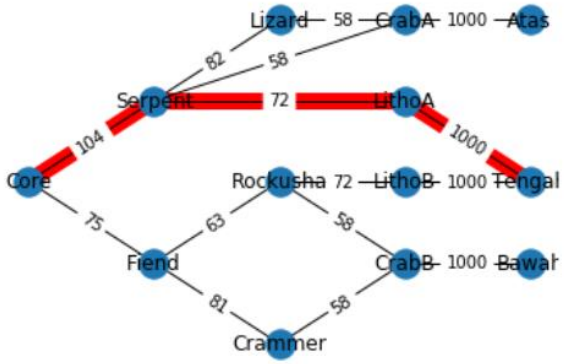
Pada uji kasus pertama, seorang *Core* ingin melakukan *ganking* pada jalur atas sambil melakukan *farming* sehingga memperoleh *gold* yang maksimal, maka pola *farming* yang diberikan oleh program sebagai berikut:



Gambar 5. Hasil pola *farming* pada jalur atas

Hasil ini sesuai dengan kejadian sebenarnya ketika seorang *Core* ingin melakukan *ganking* pada jalur atas maka pertama ia dihadapkan pada dua pilihan monster yaitu Serpent dan Fiend karena Serpent menghasilkan *gold* yang lebih besar maka *Core* akan membunuhnya. Setelah itu *Core* dihadapkan pada tiga pilihan, ingin membunuh Lizard, CrabA, atau LithoA karena Lizard memberikan *gold* yang paling besar saat itu maka *Core* akan membunuhnya kemudian membunuh CrabA dan melakukan *ganking* pada lawan di jalur atas dengan perolehan *gold* yang maksimal.

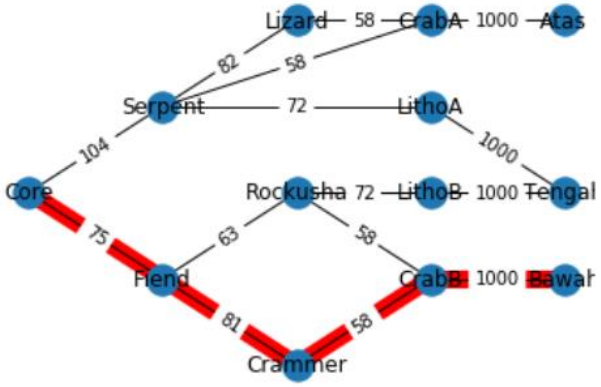
Pada uji kasus kedua, seorang *Core* ingin melakukan *ganking* pada jalur tengah sambil melakukan *farming* sehingga memperoleh *gold* yang maksimal, maka pola *farming* yang diberikan oleh program sebagai berikut:



Gambar 6. Hasil pola *farming* pada jalur tengah

Hasilnya sesuai dengan perilaku *greedy*, *Core* akan membunuh Serpent karena *gold* yang diberikan paling besar saat itu dari Fiend meskipun hasil akhirnya nanti tidak optimal. Kemudian *Core* bergerak untuk membunuh LithoA daripada Lizard karena melanggar fungsi kelayakan yaitu Lizard tidak berada di jalur *ganking* tengah sehingga dicari monster hutan dengan penghasilan *gold* terbesar lainnya.

Pada uji kasus ketiga, seorang *Core* ingin melakukan *ganking* pada jalur bawah sambil melakukan *farming* sehingga memperoleh *gold* yang maksimal, maka pola *farming* yang diberikan oleh program sebagai berikut:



Gambar 7. Hasil pola *farming* pada jalur bawah

Hasil ini sesuai dengan kejadian sebenarnya ketika seorang *Core* ingin melakukan *ganking* pada jalur bawah maka pertama ia dihadapkan pada dua pilihan monster yaitu Serpent dan Fiend. Serpent tidak dipilih karena melanggar fungsi kelayakan, yakni tidak berada di jalur *ganking* bawah sehingga dipilih penghasil *gold* terbesar yakni Fiend. Selanjutnya, *Core*

dihadapkan pada dua pilihan yaitu Rockusha dan Crammer karena Crammer menghasilkan *gold* paling besar saat itu maka *Core* akan membunuhnya kemudian membunuh CrabB dan melakukan *ganking* pada jalur bawah.

D. Perbandingan antara Simuasi dan Permainan

Program simulasi yang dibuat merupakan visualisasi dari peta permainan Mobile Legends: Bang Bang. Berikut merupakan pola *farming* pertama ketika *Core* ingin melakukan *ganking* pada jalur atas dalam permainan sebenarnya:

Gambar 8. Hasil pola *farming* pada jalur atas permainan

Gambar selanjutnya merupakan pola *farming* kedua ketika *Core* ingin melakukan *ganking* pada jalur tengah dalam permainan sebenarnya:

Gambar 9. Hasil pola *farming* pada jalur tengah permainan

Gambar terakhir merupakan pola *farming* ketiga ketika *Core* ingin melakukan *ganking* pada jalur bawah dalam permainan sebenarnya:

Gambar 10. Hasil pola *farming* pada jalur bawah permainan

Pada gambar pertama, terdapat perbedaan pada posisi koordinat Lizard dan CrabA dengan yang disimulasikan. Hal ini disebabkan oleh perbedaan koordinat sumbu x dan y yang dimasukkan pada program. Pada gambar kedua, nampak sedikit berbeda juga antara simulasi dengan permainan karena perbedaan sudut pandang *Core*, pada simulasi *Core* berada pada arah Barat sedangkan pada permainan *Core* berada pada arah Barat Daya, begitu pula dengan gambar ketiga.

Sebenarnya terdapat beberapa pertimbangan dalam melakukan *farming.* Pada permainan sebenarnya, seorang *Core* juga membutuhkan *buff*, dan pola farming yang lebih bervariasi daripada yang divisualisasikan oleh sebab itu permainan ini merupakan permainan strategi *real time.* Tidak ada pola yang benar-benar pasti dalam melakukan *ganking* dan *farming.* Seorang *Core* dalam permainan harus mampu menyesuaikan pola tersebut berdasarkan kondisi tim dan lawan, perlu banyak pengalaman dan insting yang kuat untuk memenangkan permainan secara efektif. Program simulasi yang dibuat dapat digunakan oleh pemula yang ingin belajar bagaimana pola *farming* dan *ganking* yang seorang *Core.* Penulis berharap, program simulasi ini dapat bermanfaat bagi pemula untuk menambah pengalaman dan insting bermain seorang *Core.*

IV. Kesimpulan

Algoritma *greedy* dapat digunakan pada persoalan optimasi. Persoalan tersebut sangat berkaitan erat dengan permainan strategi sehingga algoitma ini cukup populer digunakan sebagai strategi. Salah satu penerapan algoritma ini adalah pada pola *farming* berdasarkan keputusan *ganking Core* dalam permainan Mobile Legends. Algoritma *greedy* akan menentukan monster hutan yang menghasilkan *gold* terbanyak saat itu tanpa memikirkan hasil akhirnya sehingga algoritma ini bertujuan untuk memaksimalkan perolehan *gold* seorang *Core*. Simulasi yang dibuat dengan pendekatan ini diharapkan dapat membantu pemula yang ingin mengetahui pola *farming* dan *ganking* seorang *Core* dalam permainan Mobile Legends: Bang Bang.

V. Pranala Video di YouTube

https://youtu.be/LGG9PjjxBPs

VI. Ucapan Terima Kasih

Terima kasih kepada Tuhan Yang Maha Esa berkat kasih, karunia, berkat, dan limpahan rahmat-Nya, penulis dapat menyelesaikan makalah ini dengan baik. Tidak lupa, penulis juga berterima kasih kepada Bapak Rinaldi Munir selaku dosen pengajar Strategi Algoritma K4 yang telah menyediakan waktu dan tenaganya untuk menyampaikan ilmu terkait Strategi Algoritma kepada mahasiswa K4. Penulis juga berterima kasih kepada keluarga, kerabat, dan pihak-pihak lainnya atas segala doa dan dukungan yang telah diberikan.

Referensi

[1] Jarvis. 2019. Memahami Berbagai Istilah Penting dalam Permainan Mobile Legends: Bang-Bang. https://www.blibli.com/friends/blog/istilah-penting-dalam-permainan-mobile-legends-bang-bang/ diakses 6 Mei 2021 pukul 20.21 WIB.

[2] Keqing, Fans. 2020. Nama-Nama Monster Jungle Mobile Legends (ML).

https://esportsku.com/nama-nama-monster-jungle-mobile-legends-ml/ diakses 6 Mei 2021 pukul 12.32 WIB.

[3] Munir, Rinaldi. 2021. Algoritma Greedy (Bag. I dan II). Bandung https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag2.pdf diakses 6 Mei 2021 pukul 21.32 WIB.

Bandung, 26 April 2021

Rolland Steven Supardi
13519173

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.