Penerapan Algoritma Greedy Pada Game Tower Defense: Tower of Greece

Husni Munaya - 13513022[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]husni.munaya@students.itb.ac.id

***Abstrak*—Permasalahan optimisasi merupakan permasalahan yang sering kita temui di kehidupan sehari-hari. Contohnya adalah *masalah knapsack, job scheduling* dan *traveling salesman problem*. Ada banyak algoritma yang dapat menyelesaikan permasalah tersebut, salah satunya adalah algoritma greedy. Meskipun hasilnya tidak selalu optimal, algoritma greedy sangatlah efisien, karena algoritma tersebut akan mengambil keputusan tanpa memperhitungkan konsekuensi kedepannya. Pada makalah ini, akan dijelaskan bagaimana penerapan algoritma greedy dalam game *Tower* of Greece.**

***Kata kunci*—greedy, *tower* defense.**

I. PENDAHULUAN

1.1 Pengertian Tower Defense

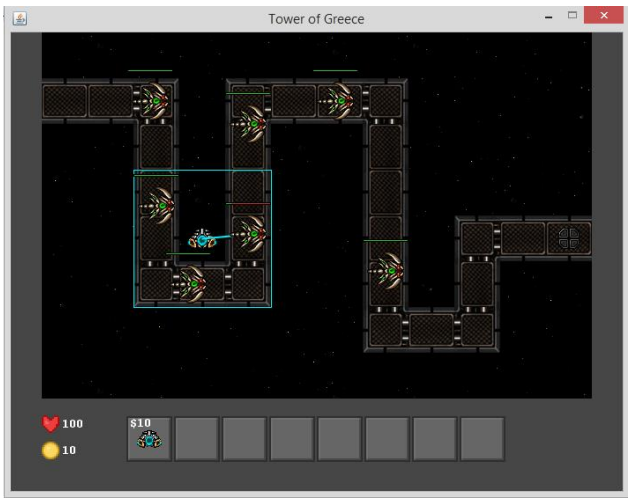
Tower Defense (TD) merupakan permainan strategi pengaturan *tower* (bangunan, senjata, dll) yang bertujuan untuk menghentikan musuh yang akan melintas. *Tower* ini akan menemba musuh dalam radiusnya. *Tower* biasanya memiliki beberapa tipe dan level dengan kemampuan, biaya pembelian, dan biaya upgrade yang berbeda. Ketika musuh dikalahkan, pemain akan mendapatkan uang sesuai dengan jenis musuhnya. Uang tersebut dapat digunakan untuk meng-upgrade *tower*.

Gambar 1. 1: Plant vs Zombie merupakan salah satu jenis game *tower defense*. Sumber: http://www.popcap.com

1.2 Tower of Greece

Tower of Greece merupakan salah satu jenis game *tower* defense yang sangat sederhana. Game ini dibuat oleh Ryan Yonata, Julio Savigny, dan Husni Munaya untuk memenuhi tugas matakuliah IF2210 Pemrograman berorientasi Objek.

Pada game ini, terdapat satu jenis musuh dan satu jenis *tower*. Pada awalnya, pemain memiliki 10 koin yang digunakan untuk membeli *tower*. Musuh akan berdatangan sesaat setelah permainan dijalankan. Musuh tersebut akan berdatangan hingga pemain berhasil menyelesaikan permainan atau kalah dari permainan. Jika musuh berhasil mencapai markas kita, nyawa dari pemain akan berkurang. Tujuan dari game ini adalah mempertahankan markas hingga pemain berhasil membunuh 20 musuh.



Gambar 1.2: Tampilan game *Tower of Greece*

1.3 Analisis Masalah

Pemilihan dan penempatan *tower* merupakan strategi yang sangat penting pada permainan ini. *Tower* harus ditempatkan sedemikian rupa sehingga bisa menghasilkan serangan terbesar terhadap musuh. Selain itu, algoritma penembakan musuh yang dilakukan oleh *tower* akan sangat mempengaruhi jalannya permainan.

Pada permainan yang akan digunakan dalam pembahasan makalah ini, penempatan dan pemilihan *tower*

akan dilakukan oleh pemain. Sehingga program tidak dapat mengoptimalkan aspek tersebut karena hal tersebut akan dikendalikan oleh pemain. Aspek yang akan kita optimalkan pada kasus ini hanyalah strategi penembakan yang dilakukan *tower* terhadap musuh sehingga kita dapat meminimalkan jumlah musuh yang masuk ke markas kita. Strategi penembakan musuh tersebut akan langsung diterapkan pada *source code* dari permainan *Tower of Greece*

II. DASAR TEORI

2.1 Algoritma Greedy

Algoritma greedy adalah algoritma yang bertujuan untuk menyelesaikan permasalahan optimisasi, yaitu maksimasi atau minimasi. Secara bahasa, *greedy* berarti serakah, rakus, tamak. Prinsip dari algoritma greedy sendiri adalah "ambil apa yang dapat kamu ambil sekarang!", tanpa memikirkan konsekuensi kedepannya. Oleh karena itu algoritma ini sangatlah efisien jika dibandingkan dengan algoritma lainnya seperti *exhaustive search, depth-first search, atau breadth-first search*, meskipun solusi yang didapat tidak akan selalu optimal.

Algoritma *greedy* akan menyelesaikan masalah langkah demi langkah, oleh karena itu, perlu dilakukan keputusan terbaik dalam menentukan pilihan. Pada setiap langkahnya, algoritma greedy akan melakukan hal berikut:

1. Mengambil pilihan terbaik yang dapat diperoleh saat itu tanpa memperhitungkan konsekuensi kedepannya.
2. Berharap bahwa pemilihan optimum lokal (terbaik) pada setiap langkah akan membawa hasil dengan yang optimal secara global.

2.2 Elemen Algoritma Greedy

Algoritma greedy memiliki beberapa elemen, yaitu:

1. Himpunan kandidat
   Himpunan ini berisi kandidat yang dapat menjadi solusi permasalahan. Setiap elemen akan diambil dari himpunan kandidat pada setiap langkahnya.
2. Himpunan solusi
   Himpunan solusi merupakan himpunan kandidat yang terpilih sebagai solusi. Himpunan ini merupakan himpunan bagian dari himpunan kandidat.
3. Fungsi seleksi
   Fungsi seleksi merupakan fungsi yang digunakan untuk memilih kandidat agar menghasilkan solusi yang optimal.
4. Fungsi kelayakan.
   Funsi kelayakan merupakan fungsi yang memeriksa apakah suatu kandidat yang telah terpilih layak untuk menjadi solusi.
5. Fungsi Objektif
   Fungsi objektif merupakan fungsi yang akan memilih solusi yang paling optimal.

Mari kita lihat persoalan penukaran uang dibawah ini. Kita akan menyelesaikannya dengan algoritma greedy.

*Diberikan uang senilai A. Tukar A dengan koin-koin yang yang ada. Berapa jumlah **minimum** koin yang diperlukan untuk penukaran tersebut?* A = 32 dan A = 30, koin yang tersedia: 1, 5, 10, 25.

Dari soal di atas, kita dapat mengetahui bahwa permasalahan tersebut tergolong dalam permasalahan minimisasi. Strategi greedy yang akan kita lakukan adalah memilih koin dengan nilai terbesar dari himpunan koin yang tersisa.

Elemen untuk masalah penukaran uang:

1. Himpunan kandidat: himpunan koin yang merepresentasikan nilai 1, 5, 10, 25.
2. Himpunan solusi: total nilai dari koin yang dipilih sama jumlahya dengan nilai uang yang ditukarkan.
3. Fungsi seleksi: pilihkan koin yang bernilai tinggi dari himpunan kandidat yang tersisa.
4. Fungsi kelayakan: memeriksa apakah nilai total dari himpunan koin yang dipilih tidak melebihi jumlah uang yang harus dibayar.
5. Fungsi objektif: jumlah koin yang didapatkan minimum.

Langkah penyelesaian untuk A = 32:

1. Pilih 1 buah koin 25. (total 25)
2. Pilih 1 buah koin 5. (total 30)
3. Pilih 2 buah koin 1. (total 32)

Solusi: jumlah koin minimum = 4 (solusi optimal)

Langkah penyelesaian untuk A = 30:

1. Pilih 1 buah koin 25. (total 25)
2. Pilih 1 buah koin 5. (total 30)

Solusi: jumlah koin minimun = 2 (solusi optimal)

2.1 Skema Umum Algoritma Greedy

Berikut merupakan gambaran umum dari algoritma greedy, ditulis dalam *pseudo code.*

```
function greedy(input C: himpunan_kandidat)→ himpunan_kandidat
{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy
  Masukan: himpunan kandidat C
  Keluaran: himpunan solusi yang bertipe himpunan_kandidat
}
Deklarasi
    x : kandidat
    S : himpunan_kandidat

Algoritma:
      S ← {}   { inisialisasi S dengan kosong }
      while (not SOLUSI(S)) and (C ≠ {} ) do
       x ← SELEKSI(C)              {      pilih sebuah kandidat dari C}
       C ← C - {x}            { elemen himpunan kandidat berkurang satu }
      if LAYAK(S ∪ {x}) then
                S ← S ∪ {x}
         endif
     endwhile
{SOLUSI(S) or C = {} }
```

```
if SOLUSI(S) then
   return S
else
   write('tidak ada solusi')
endif
```

Pada akhir setiap lelaran, solusi yang terbentuk adalah optimum lokal. Pada akhir kalang while-do diperoleh optimum global.

Perlu diperhatikan bahwa hasil yang didapat dari algoritma greedy belum tentu merupakan solusi optimal, melainkan solusi sub-optimal atau *pseudo*-optimal. Ada beberapa alasan mengapa algoritma greedy belum tentu menghasilkan solusi optimal, yaitu sebagai berikut:

1. Algoritma greedy tidak beroprasi secara menyeluruh terhadap semua solusi alternatif yang ada sebagaimana pada metode *exhaustive search.*
2. Terdapat beberapa fungsi seleksi yang berbeda, sehingga kita harus memilih fungsi yang tepat jika kita menginginkan solusi yang optimal.

Jadi, pada sebagian masalah, algoritma greedy tidak selalu memberikan solusi yang optimal. Contoh dari masalah yang tidak memberikan solusi optimal ketika menggunakan algoritma greedy adalah masalah persoalan penukaran uang yang akan dicontohkan sebagai berikut.

Dari soal di atas, kita dapat mengetahui bahwa permasalahan tersebut tergolong dalam permasalahan minimisasi. Strategi greedy yang akan kita lakukan adalah memilih koin dengan nilai terbesar dari himpunan koin yang tersisa.

Contoh 1
Jumlah uang yang ingin ditukar = 7
Koin: 5, 4, 3, 1
Solusi greedy: 5, 1, 1 (3 koin. Tidak optimal)
Solusi optimal: 4, 3 (2 koin)

Contoh 2
Jumlah uang yang ingin ditukar = 15
Koin: 10, 7, 1
Solusi greedy: 10, 1, 1, 1, 1, 1 (6 koin. Tidak optimal)
Solusi optimal: 7, 7, 1 (3 koin)

Contoh 3
Jumlah uang yang ingin ditukar = 20
Koin: 15, 10, 1
Solusi greedy: 15, 1, 1, 1, 1, 1 (6 koin. Tidak optimal)
Solusi optimal: 10, 10 (2 koin)

Contoh 4
Jumlah uang yang ingin ditukar = 16
Koin 10, 8, 2
Solusi greedy: 10, 2, 2, 2 (4 koin. Tidak optimal)
Solusi optimal: 8, 8 (2 koin)

III. PENERAPAN ALGORITMA GREEDY

Dalam permainan ini, hanya terdapat satu jenis *tower* dan satu jenis musuh. Selama permainan berlangsung, musuh akan muncul dengan waktu dan kecepatan yang konstan. *Tower* dapat menyerang musuh jika musuh berada dalam area serang. Pemain akan menang jika jumlah musuh yang sudah dibunuh mencapai 20. Pemain memiliki nyawa sebesar 100. Ketika musuh berhasil memasuki markas, nyawa pemai akan berkurang 10.

Tabel di bawah ini menggambarkan spesifikasi yang dimiliki oleh tiap komponen dalam permainan *Tower of Greece.*

1. *Tower*

Damage	2
Area serang	104
Harga	10

Tabel 3.1: Spesifikasi *tower*

2. Creep (musuh)

Nyawa	52
Damage	10
Bounty	5

Tabel 3.2: Spesifikasi creep (musuh)

3. Player

Nyawa	100
Koin	10

Tabel 3.3: Spesifikasi player

Strategi yang harus diterapkan pada permainan ini adalah bagaimana mekanisme penembakan musuh yang dilakukan oleh *tower* sehingga jumlah musuh yang memasuki markas adalah seminimal mungkin. Terdapat beberapa strategi yang dapat diaplikasikan, yaitu sebagai berikut.

3.1 Strategi Greedy 1

Pada strategi greedy 1, algoritma ini akan menembak musuh yang baru saja memasuki area serang dari *tower*. Jika *tower* sedang menembak musuh, namun terdapat musuh baru yang masuk ke area serang, *tower* akan mengganti targetnya kepada musuh yang baru masuk tersebut.

Gambar 3.1.1: Tower akan mengganti sasaran tembaknya begitu terdapat musuh yang baru memasuki area tembak. (musuh bergerak dari kiri ke kanan)

3.1.1 Elemen Algoritma Greedy 1

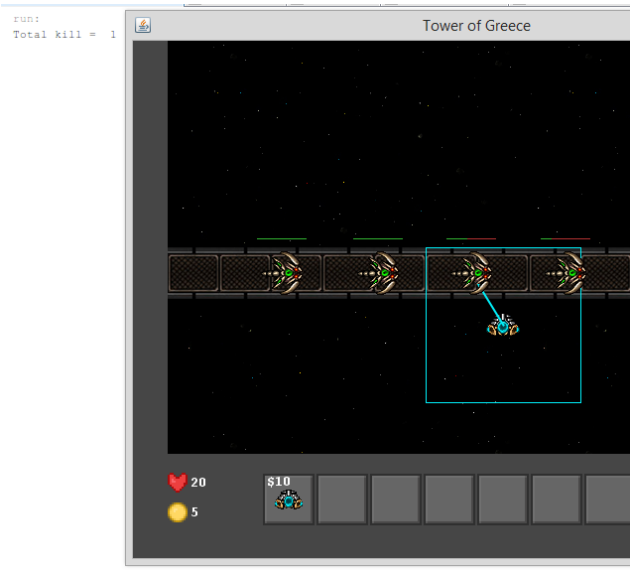
1. Himpunan Kandidat
   Himpunan kandidat dalam permasalahan ini adalah semua musuh yang berada dalam permainan.
2. Himpunan Solusi
   Himpunan solusi dalam permasalahan ini adalah himpunan bagian dari kandidat yang menyebabkan pemain bisa memenangkan permainan.
3. Fungsi Seleksi
   Fungsi seleksi dari algoritma greedy 1 adalah memilih musuh yang paling baru memasuki area tembak.
4. Fungsi Kelayakan
   Memeriksa apakah nyawa dari pemain masih lebih dari 0.
5. Fungsi Objektif
   Fungsi objektifnya adalah memenangkan permainan dan meminimalkan jumlah musuh yang memasuki markas.

3.1.2 Pseudocode Algoritma Greedy 1

```
// Global variable
integer shotCreep = -1 // -1 berarti tower
tidak  menembak creep manapun
boolean shooting  = false

procedure shoot() {
  shooting = false
  for var i = 1 to creeps.size() {
    sortestDistance =
    if (this.towerArea.intersect(creep[i])) {
      shooting = false
      shotCreep = i
    }
  }
}
```

3.1.3 Analisis Algoritma Greedy 1

Dari parameter yang sudah disebutkan di awal bab ini, didapatkan hasil dari percobaan sebagai berikut.



Gambar 3.1.3.1: Jalannya permainan dengan algoritma greedy 1.

Jumlah *tower* yang digunakan	2
Jumlah musuh yang berhasil masuk ke markas	10 (kalah dari permainan)
Jumlah musuh yang berhasil di bunuh:	3

3.2 Strategi Greedy 2

Pada strategi greedy 2, musuh yang akan dipilih adalah yang paling pertama memasuki area serang *tower*. *Tower* akan menembak musuh tersebut hingga musuh keluar dari serangan *tower*. Ketika musuh yang sedang ditembak sudah berada di luar serangan *tower*, algoritma ini akan mencari musuh paling pertama selanjutnya dan menembaknya hingga keluar dari area serang. Artinya, *tower* ini tidak akan melepaskan sasaran tembaknya hingga musuh tersebut keluar dari area.

Gambar 3.2.1: Meskipun terdapat musuh baru yang memasuki area *tower* akan tetap terus menyerang musuh pertama hingga musuh tersebut berada di luar area serang. (musuh bergerak dari kiri ke kanan)

3.2.1 Elemen Algoritma Greedy 2

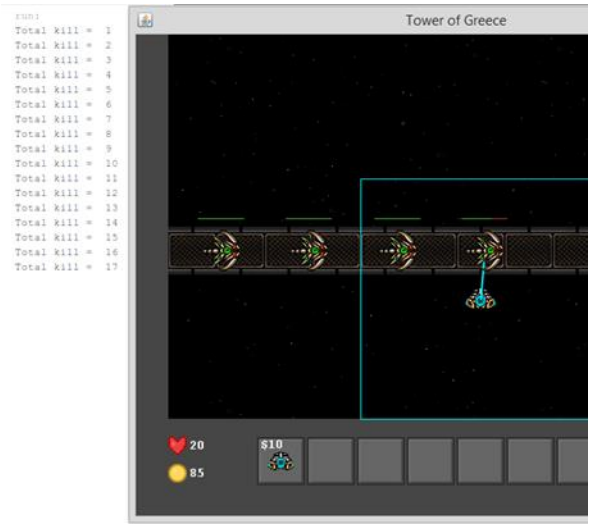
1. Himpunan Kandidat
   Himpunan kandidat dalam permasalahan ini adalah semua musuh yang berada dalam permainan.
2. Himpunan Solusi
   Himpunan solusi dalam permasalahan ini adalah himpunan bagian dari kandidat yang menyebabkan pemain bisa memenangkan permainan.
3. Fungsi Seleksi
   Fungsi seleksi dari algoritma greedy by position adalah memilih musuh yang pertama kali memasuki area.
4. Fungsi Kelayakan
   Memeriksa apakah nyawa dari pemain masih lebih dari 0.
5. Fungsi Objektif
   Fungsi objektfnya adalah memenangkan permainan dan meminimalkan jumlah musuh yang memasuki markas.

3.2.2 Pseudocode Algoritma Greedy 2

```
// Global variable
integer shotCreep = -1 // -1 berarti tower
tidak  menembak creep manapun
boolean shooting  = false

procedure shoot() {

  if (shotCreep <> -1 and
this.towerArea.intersect(shotCreep)) {
    shooting = true
  } else {
    shooting = false
  }

  if (!shooting) {
    for var i = 1 to creep.size() {
      if (this.towerArea.intersect(creep[i])){
        shooting = true
        shotCreep = i
      }
    }
  }

}
```

3.2.3 Analisis Algoritma Greedy 2

Dari parameter yang sudah disebutkan di awal bab ini, didapatkan hasil dari percobaan sebagai berikut.



Gambar 3.2.3.1: Jalannya permainan dengan algoritma greedy 2.

Jumlah *tower* yang dibutuhkan untuk memenangkan permainan	1
Jumlah musuh yang berhasil masuk ke markas	9
Jumlah musuh yang berhasil di bunuh:	20 (berhasil memenangkan permainan)

Dengan algoritma greedy 2, pemain dapat memenangkan permainan walaupun hanya menggunakan 1 tower. Namun jumlah musuh yang berhasil masuk cukup banyak, yaitu 9.

IV. KESIMPULAN

1. Algoritma greedy merupakan algoritma yang dapat digunakan pada persoalan optimasi. Algoritma ini sangatlah efisien meskipun hasil yang didapatkan bukanlah solusi optimal, melainkan solusi sub-optimal atau solusi *pseudo*-optimal.
2. Pada persoalan ini, algoritma greedy 2 lebih tepat untuk digunakan karena dapat memberikan solusi yang optimal. Dengan algoritma ini, pemain dapat memenangkan permainan *Tower* of Greece dengan lebih mudah.
3. Penerapan algoritma yang dibahas pada makalah ini masih perlu diperbaiki, sebab meskipun pemain dapat memenangkan permainan, jumlah musuh yang masuk ke markas masih cukup banyak.

V. APENDIKS

Source code dari permainan Tower of Greece dapat di akses di https://github.com/jsavigny/OOP_TowerDefense/

VI. UCAPAN TERIMA KASIH

Saya mengucapkan terimakasih kepada Tuhan Yang Maha Esa karena atas rahmat-Nya makalah ini dapat diselesaikan. Tidak lupa saya mengucapkan terimakasih kepada Pak Rinaldi dan Bu Ulfa atas ilmu dan bimbingannya dalam kuliah IF2211 Strategi Algoritma. Saya juga ingin mengucapkan terimakasih kepada teman-teman atas dukungan yang diberikan pada penulisan makalah ini.

VII. REFERENSI

[1] Rinaldi Munir, Diktat Kuliah IF3051 Strategi Algoritma, Bandung: Program Studi Teknik Informatika ITB, 2009.
[2] https://www.cs.rochester.edu/~gildea/csc282/slides/C16-greedy.pdf diakses pada tanggal 04/05/2015 21.52

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, 10 December 2014

Husni Munaya -13513022