Penerapan Algoritma Greedy pada Bot AI Permainan *Bang!*

Nugroho Satrijandi - 13510032
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13510032@std.stei.itb.ac.id

Abstraksi*—Di zaman dengan kemajuan teknologi yang sangat pesat ini, banyak permainan kartu yang dijadikan permainan di *mobile* maupun *desktop*. Salah satu dari permainan kartu tersebut adalah *Bang!*. Menarik bila kita melihat bagaimana cara Artificial Intelligent bermain permainan ini untuk mencapai kemenangan. Pada tulisan ini akan dijelaskan sedikit tentang gameplay permainan *Bang!* ini, serta strategi yang harus digunakan untuk dapat memenangkan permainan ini. Tulisan ini juga akan menjelaskan tentang penerapan algoritma Greedy pada AI permainan *Bang!

Kata Kunci*— *card game, Bang!, Greedy, AI

I. PENDAHULUAN

Permainan kartu secara fisik yang memiliki inovasi dari *gameplay* dalam beberapa tahun terakhir banyak bermunculan, seperti YugiOH, PetWars, Saboteur, UNO, Monopoly Deal, *Bang!*, dan masih banyak lagi. Namun perkembangan teknologi yang sangat pesat terutama perkembangan permainan elektronik pada *mobile* atau *desktop* membuat banyak permainan kartu tersebut dibuat menjadi suatu aplikasi permainan pada *mobile*, *dekstop*, dan juga *web*. Salah satu dari permainan kartu tersebut adalah permainan *Bang!*.

Gambar 1 Cover Permainan *Bang!*

Bang! merupakan permainan kartu bertemakan "*wild-west*"(*cowboy).* Permainan ini dirancang oleh Emiliano Sciarra dan di-*publish* oleh daVinci Editrice di tahun 2002. Permainan ini dimainkan oleh 4-7 pemain dimana masing-masing pemain akan mendapatkan peran tertentu:

- *Sheriff*, (x1)
- *Deputy Sheriff,* (x0, x1 or x2)
- *Outlaw*, (x2 or x3)
- *Renegade*, (x1, x2 dengan adanya tambahan)

Masing-masing pemain akan mendapatkan karakter unik dengan kemampuan spesial dan dengan "*peluru*" sebagai *life points.* Objektif dari permainan ini berbeda untuk masing-masing peran.

- *Sheriff* dan *Deputy Sheriff* harus membunuh *Outlaw* dan *Renegade.*
- *Outlaw* harus membunuh *Sheriff.*
- *Renegade* harus menjadi karakter terakhir di dalam permainan. Karakter ini harus membunuh semua karakter lain dengan *Sheriff* karakter terakhir yang harus dibunuh.

Kartu peran tiap pemain ditutup, hanya pemain itu yang tahu apa karakter yang dimilikinya, kecuali Sheriff. Namun Sheriff mendapatkan tambahan maksimum *life points* sebanyak 1 buah.

Pada awal permainan setiap pemain dibagikan kartu sebanyak jumlah "*peluru*" yang dimilikinya. Sheriff memiliki hak untuk mengambil giliran pertama. Tiap pemain memiliki 3 fase.

1. *Draw* 2 kartu
2. Memainkan sejumlah kartu yang ada di tangan.
    - Pemain boleh menolong dirinya sendiri, orang lain, atau melukai untuk mengeliminasi pemain lain. Pemain tidak dipaksa untuk mengeluarkan kartu pada fase ini tidak ada batasan jumlah kartu yang boleh dikeluarkan, selain batasan:
        i. Hanya boleh mengeluarkan kartu *BANG!* setiap giliran, kecuali ada efek yang membuat player tersebut dapat mengeluarkan lebih dari 1 kartu(akan dijelaskan kemudian).
        ii. Tidak boleh ada 2 kartu identik di atas halaman permainan pemain tersebut.

    Pemain memiliki hanya boleh memiliki satu macam senjata pada satu waktu. Jika ada kartu senjata lain, maka kartu tersebut harus di-*discard.* Secara normal sebuah kartu memiliki

efek tertentu yang lagsung digunakan dan di-*discard.* Ketika suatu kartu dikeluarkan hanya pada saat giliran pemain tersebut, akan tetapi ada pengecualian pada kartu *Beer*, *Bang!* dan *Missed*. Efek tiap kartu bertahan sampai kartu itu di-*discard*(kartu *Cat Balou* atau *Panico*) atau ada kejadian khusus terjadi seperti (kasus efek kartu *Jail* dan *Dynamite*).

3. Membuang kartu yang berlebih. Kartu di tangan pemain harus lebih kecil sama dengan *life-points* yang dimilikinya. Jika lebih, maka pemain bebas menentukan kartu apa saja yang ingin dibuang.

Ada beberapa konsep dari permainan Bang ini secara umum, di antaranya:

- Jarak(*Distance*): didefinisikan sebagai jarak terkecil antar pemain.
- Serangan : Kartu dengan tanda *Bang!* digunakan untuk menyerang pemain lain dengan mengurangi *life-points*.
- Kartu Bang hanya dapat digunakan untuk menyerang pemain lain satu kali dengan jarak(*range*) sesuai kemampuan kartu senjata. Kartu *Bang!* dapat dikeluarkan lebih dari 1 kali jika ada kartu senjata Volcanic.
- Kartu senjata dapat memiliki jarak(*range*) yang berbeda-beda.
- Menghindari serangan dapat menggunakan kartu "*Missed*", jika tidak maka *life-points* akan berkurang.
- Ketika "peluru"(*life-points*) pemain habis, maka pemain keluar dari permainan.
- Pemain dapat menambahkan "peluru" nya kembali dengan cara mengeluarkan kartu "Beer". Kartu ini juga dapat digunakan untuk mencegah peluru berkurang dari 0 menjadi 1.

Selain itu ada beberapa aturan pelanggaran dan hadiah. Jika seorang *Sheriff* membunuh *Deputy*-nya maka seluruh kartu yang dimilikinya harus dibuang. Jika seorang pemain membunuh seorang *Outlaw*, maka pemain tersebut dapat mengambil 3 kartu tambahan dari tumpukan kartu.

II. Dasar Teori Algoritma Greedy

Algoritma *Greedy* umumnya digunakan untuk memecahkan masalah optimasi. Dalam menyelesaikan persoalan optimasi, *Greedy* mencari persoalan optimasi maksimasi dan minimasi yang paling optimum. Prinsip dari algoritma *Greedy* yaitu "*take what you can get now!*". Algoritma *Greedy* membentuk sebuah solusi langkah per langkah. Pada setiap langkah yang dilalui terdapat banyak pilihan yang perlu dieksplorasi. Untuk itu pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan. Algoritma *Greedy* adalah algoritma yang memecahkan masalah langkah per langkah; pada setiap langkahnya:

1. mengambil langkah yang terbaik yang bisa diperoleh pada saat itu tanpa memperhitungkan konsekuensi dari pengambilan keputusan itu ke depannya. (prinsip "*take what you can get now!*")
2. berharap bahwa dengan memilih optimum lokal pada setiap langkahnya akan berakhir dengan optimum global.

Ada beberapa elemen pada algoritma *Greedy*, antara lain:

1. Himpunan Kandidat, C
2. Himpunan Solusi, S
3. Fungsi Seleksi (*selection function*)
4. Fungsi Kelayakan (*feasible*)
5. Fungsi Objektif

Dapat dikatakan, algoritma *Greedy* melibatkan pencarian sebuah solusi himpunan S dari himpunan kandidat C. S harus dapat memenuhi beberapa kriteria yang ditentukan yaitu menyatakan suatu solusi dan S dioptimalisasi oleh suatu fungsi objektif.

Skema umum algoritma *Greedy* adalah sebagai berikut:

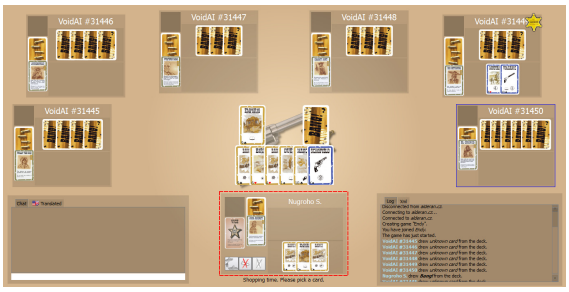
```
function greedy(input C :
    himpunan_kandidat)→himpunan_kandidat
{ Mengembalikan solusi dari persoalan optimasi
    dengan algoritma greedy
  Masukan: himpunan kandidat C
  Keluaran: himpunan solusi yang bertipe
    himpunan_kandidat
}
Deklarasi
    x : kandidat
    S : himpunan_kandidat

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
```

III. Penerapan Algoritma Greedy pada Bot Permainan *Bang!*

Pada permainan *Bang!* terdapat Bot AI yang dapat bermain seperti layaknya manusia. Berikut gambar screenshot dari sebuah permainan *Bang!* Online yang melibatkan Bot di dalamnya.



Gambar 2 Screenshot Permainan *Bang!* melibatkan Bot

Pada permainan tersebut Bot AI bermain layaknya seperti seorang manusia. Dari proses analisis dengan bermain permainan *Bang!* ini, dapat dilihat bahwa Bot AI ini menggunakan algoritma yang memiliki dasar algoritma Greedy untuk menentukan urutan kartu yang harus dikeluarkan pada setiap gilirannya. Bot AI tidak dapat mengetahui siapa teman satu tim nya, misalnya *Outlaw* tidak tahu siapa *Outlaw* lainnya, dan *Sheriff* tidak tahu siapa *Deputy* yang membantu dirinya. Yang diketahui adalah pemain yang berperan sebagai *Sheriff*.

Berikut pemetaan Bot AI setiap gilirannya pada permainan *Bang!* ke dalam elemen yang ada pada algoritma Greedy.

1. Himpunan Kandidat, C

   Pada permainan ini setiap pemain mendapatkan giliran untuk mengeluarkan kartunya masing-masing. Pada setiap giliran, masing-masing pemain dapat menggunakan beberapa atau bahkan semua kartu yang dipunyai.

   Dari sini kita dapat melihat bahwa himpunan kandidat, C, pada saat giliran pemain tersebut, terdapat beberapa kartu di tangan pemain yang di keluarkan. Kartu yang ada di tangan pemain inilah yang menjadi himpunan kandidat untuk melakukan suatu aksi.

2. Himpunan Solusi, S

   Pada permainan *Bang!* ini terdapat himpunan solusi yang berupa langkah yang dilakukan pemain atau dapat juga berupa kartu-kartu yang dapat dikeluarkan oleh pemain. Namun akan lebih baik jika urutan pengeluaran kartu-kartu tersebut diperhitungkan, karena masing-masing kartu memiliki beberapa teknik yang benar dalam mengeluarkannya.

3. Fungsi Seleksi (*selection function*)

   Fungsi seleksi yang digunakan dalam permainan ini dapat beraneka ragam. Urutan kartu yang dipilih untuk setiap giliran, dapat dikatakan tergantung dari strategi permainan. Fungsi seleksi yaitu fungsi memilih dari himpunan kandidat. Pada permainan ini fungsi seleksi yaitu memilih salah satu dari kartu-kartu yang ada di tangan. Pemilihan kartu ini berdasarkan urutan pengeluaran, dapat digunakan misalnya menggunakan kartu *action defense* terlebih dahulu sebelum mengeluarkan kartu *action attack*, atau dapat juga sebaliknya.

   Kartu yang merupakan *action defense/passive*, antara lain:

   - *Beer*, menambahkan 1 buah *life-point* pada player tersebut.
   - *Weapon Card*, merupakan kartu *equipment* yang digunakan untuk menambahkan kemampuan senjata, baik range atau *Volcanic* (dapat mengeluarkan *Bang!* lebih dari 1 kali)
   - *Emporio*, membuka kartu sebanyak pemain yang masih hidup. Pemain yang mengeluarkan kartu ini mengambil kartu pertama kali (Pada Bot AI pengambilan kartu dilakukan secara random.)
   - *Saloon*, menambahkan semua life-points setiap pemain sebanyak 1 buah.
   - *Mustang*, menambah jarak pandang musuh terhadap pemain tersebut, membuat musuh melihat pemain tersebut dengan range +1.
   - *Appaloosa*, mengurangi jarak pandang pemain tersebut terhadap musuh sebanyak -1.
   - *Manciatto*, kartu yang hanya dapat dikeluarkan saat pemain tersebut terkena serangan *Bang!* atau Gatling.
   - *Dynamite*, kartu yang dapat meledak jika pada saat pemain mengambil kartu(*draw* 2 buah kartu) dari deck.

   Sedangkan kartu yang merupakan action attack, antara lain:

   - *Bang!,* kartu yang digunakan untuk menembak musuh lain dan mengurangi *life-point* musuh sebanyak 1 buah.
   - *Panico,* mengambil 1 kartu dari tangan orang lain dengan jarak pandang 1.
   - *Cat Balou,* men-*discard* kartu pemain lain, baik yang ada di *field* atau yang ada di tangan pemain lain.
   - *Indiana,* mengurangi life-point semua pemain sebanyak 1 buah. Pemain dapat menghindari efek kartu ini dengan mengeluarkan *Bang!*.
   - *Gatling,* memiliki efek yang sama seperti *Indiana* namun untuk menghindari efek kartu ini dengan mengeluarkan *Manciatto.*
   - *Duel,* mengajak salah satu lawan beradu, lawan harus melawan dengan mengeluarkan *Bang!,* kedua pemain yang sedang beradu Bang! mengeluarkan kartu Bang! sampai salah satu pemain kehabisan kartu *Bang!*. Pemain yang kalah dalam *Bang!* atau yang menolak untuk berduel akan mengurangi *life-point* miliknya sebanyak 1 buah.

- *Jail,* memasangkan kartu ini ke pemain lain akan mengakibatkan pemain tersebut harus mengambil 1 buah kartu hati pada saat gilirannya untuk dapat menghilangkan kartu ini, jika tidak giliran pemain ini akan dilewat(*skip*). Hanya *Sheriff* yang tidak dapat dipenjara.

4. Fungsi Kelayakan (*feasible*)

   Fungsi kelayakan di sini artinya, apakah kartu tersebut dapat dikeluarkan untuk dimainkan atau tidak. Fungsi ini akan mengembalikan nilai true jika kartu yang diseleksi layak untuk dimainkan atau dapat dikeluarkan.

   Parameter layak seperti layaknya peraturan permainan.

5. Fungsi Objektif

   Banyaknya kartu di tangan pemain tidak melebihi jumlah *life-points.* Jika lebih, maka kartu yang tersisa di himpunan kandidat akan di-*discard.*

Jika dijelaskan secara luas, maka algoritma *greedy* pada Bot AI permainan bang dapat diilustrasikan seperti ini pada *pseudo-code* berikut ini.

```
  function greedyBot(input C : kartu_di_tangan) → langkah_kartu_yang_dikeluarkan
  {     Mengembalikan langkah-langkah kartu yang harus
        dikeluarkan saat giliran pemain
        pada permainan Bang!
        Masukan: kartu_di_tangan
        Keluaran: langkah_kartu_yang_dikeluarkan
  }
  Deklarasi
      x : kandidat_kartu
      S : langkah_kartu_yang_dikeluarkan

  Algoritma:
  S ← {}   { inisialisasi S dengan kosong }
  while(C != {} ) do
        x ← SELECTCARD(C){ pilih sebuah kandidat dari C}
        C ← C - {x}{ elemen himpunan kandidat berkurang satu }
        if LAYAK(x) then
                S ← S ∪ {x}
        endif
  endwhile
  {kondisi C = {}}
  if SOLUSI(S) then
        return S;
  else
        return NULL;
endif
```

Berikut akan dijelaskan fungsi `SELECTCARD(C)`, dan fungsi kelayakan `LAYAK(x)`.

Fungsi `SELECTCARD(C)` adalah fungsi untuk memilih kartu (mengutamakan kartu action attack terlebih dahulu) yang akan dicek kelayakannya pertama kali. Fungsi kelayakan setiap kartu berbeda satu dengan yang lain. Berikut pseudocode fungsi `SELECTCARD(C)`.

```
function SELECTCARD(List of Card : C) → kartu_yang_dipilih
{   Mengembalikan list kartu yang dipilih untuk dimainkan
    Masukan : List kartu yang ada di tangan
    Keluaran : Kartu-kartu yang dipilih untuk dimainkan
}

Deklarasi:
    found : boolean
    card : Card

Algoritma:
    found ← false;
    for each card in C do
        if isAttack(card)
            return card;
        endif
    endfor {jika setelah iterasi tidak ada kartu yang merupakan attack card}
    {sisa kartu merupakan kartu pasif}
    return first_card;
```

Di bawah ini pseudocode fungsi `LAYAK(x)` untuk mengecek kelayakan suatu kartu untuk dikeluarkan.

Fungsi `LAYAK(X)` merupakan fungsi yang masih sangat sederhana tanpa ada pengetahuan intelegensia lain.

```
function LAYAK(Card : X) ? boolean
{   Mengembalikan true jika kartu X dapat dikeluarkan dan dapat dikeluarkan atau dimainkan
    Masukan : Kartu bebas
    Keluaran : True jika kartu dapat dikeluarkan atau dimainkan, False jika kartu tidak dapat dikeluarkan
}

Deklarasi:

Algoritma:
depend on X
    Beer : return true;
    Weapon : return true;
    Saloon : return true;
    Mustang : return true;
    Appaloosa : return true;
    Dynamite : return true;
    Bang! : if(BangAllowed){
                target = random();
                if(TargetReachable(target)){
                    setTarget(target);
                    return true;
                }else{
                    return false;
                }
            else{
                return false;
            }
    Panico :
             target = random();
             if(TargetReachable(target)){
                    setTarget(target);
                    return true;
            }else{
                return false;
            }
    Cat Balou : target = searchTarget();
                return true;
    Indiana :   return true;
    Jail : targetRandom = random();
           If(targetRandom!=Sheriff){
               setTarget(target);
              return true;
           else{
               return false;
           }
    Gatling : return true;
    Duel : return true;
    Manciatto : return false;
    Emporio : return true;
```

Fungsi `SOLUSI(S)` akan mengembalikan true jika S tidak kosong, dan sebaliknya menghasilkan false jika S kosong dengan kata lain tidak ada solusi yang dihasilkan.

Dari algoritma Greedy untuk masalah pencarian kartu-kartu apa saja yang dikeluarkan saat giliran pemain tersebut akan dianalisis pada bab selanjutnya.

IV. Analisis Penerapan Algoritma Greedy pada Permainan Bang!

Dari Algoritma Greedy di atas, muncul beberapa analisis efektifitas dari algoritma tersebut.

Algoritma Greedy pada bab 3, Bot AI memiliki kemampuan kecerdasan komputer yang cukup rendah. Pada algoritma Greedy permainan ini, terkadang Bot AI hanya melakukan semacam randomisasi pada saat memainkan kartu Bang!, Panico, atau Cat Balou.

Pada algoritma ini dilihat bahwa terkadang Bot AI bertindak seperti pemain bodoh, dengan melakukan serangan ke sembarang pemain. Kasus ini dapat diasah lebih lagi dengan menggunakan tambahan pengetahuan kepada Bot AI agar lebih cerdas dalam bermain. Pengetauan seperti suspek musuh, atau teman.

Dalam permainan kartu asli Bang! ini setiap pemain (manusia), memiliki teknik tersendiri untuk menebak siapakah musuh yang dihadapi atau teman yang membantu dirinya untuk menang. Hal ini biasanya ditunjukkan dengan Outlaw biasanya akan menyerang *Sheriff* terlebih dahulu, atau *Deputy* yang biasanya akan menolong Sheriff dengan cara tidak menembak *Sheriff,* membunuh *Outlaw* ataupun dengan menambahkan *life-point* dari *Sheriff* tersebut. Pengetahuan tersebut harus ditambahan dalam algoritma program.

Atau dapat juga pada kartu-kartu tertentu seperti *Beer*,

Jika ditambahkan suatu pengetahuan atau suatu kecerdasan, maka AI Bot akan berlaku seperti pada algoritma program di bawah ini, saat pengecekan dengan fungsi `LAYAK_PLUS(X)`.

```
function LAYAK_PLUS(Card : X) ? boolean
{   Mengembalikan true jika kartu X dapat
dikeluarkan dan dapat dikeluarkan atau dimainkan
    dengan adanya tambahan kecerdasan pada Bot
AI.
    Masukan : Kartu bebas
Keluaran : True jika kartu dapat dikeluarkan
atau dimainkan, False jika kartu tidak dapat
dikeluarkan
}

Deklarasi:

Algoritma:
depend on X
    Beer : if(life_full){
                return false;
           }else{
                return true;
           }
    Weapon : if(Weapon_type = Volcanic){
                return true;
             }else{
                if (current_range < Weapon){
                    return true;
                }
             }
    Saloon : depend on role:
             Outlaw : return false;
             Deputy : return true;
             Sheriff : return true;
             Renegade : return true;
    Mustang : if(Mustang Card Already Exist){
                return true;
              }else{
                return false;
              }
    Appaloosa : if(Appaloosa Card Already
Exist){
                    return true;
                }else{
                    return false;
                }
    Dynamite : if(life_point <= 2){
                    return true;
               }else{
                    return false;
                }
    Bang! : if(BangAllowed){
                target = searchTarget();
{mencari siapa target yang akan di-Bang}
                if(TargetReachable(target)){
                    setTarget(target);
                    return true;
                }else{
                    return false;
                }
            else{
                return false;
            }
    Panico :
             target = searchTarget();
             if(TargetReachable(target)){
                    setTarget(target);
                    return true;
            }else{
                return false;
            }
    Cat Balou : target = searchTarget();
                return true;
    Indiana :   if(Sheriff_health < 3){
                    depend on Role:
                    Deputy : return false;
                    Outlaw : return true;
                    Renegade : return true;
                    Sheriff : return true;
                }else{
                    return true;
                }
    Gatling :       depend on Role:
                            Deputy :
        if(Sheriff_health < 3){return false;
                }else{
                    return true;
                } default : return true;

    Duel : if(Sheriff_health < 3){
                    depend on Role:
                    Deputy : return false;
                    Outlaw : return true;
                    Renegade : return true;
                    Sheriff : return true;
                }else{
                    return true;
                }
    Gatling :       depend on Role:
                            Deputy :
        if(Sheriff_health < 3){return false;
                }else{
                    return true;
                } default : return true;
    Jail : targetRandom = random();
           If(targetRandom!=Sheriff){
               setTarget(target);
              return true;
           else{
               return false;
           }

    Manciatto : return false;
    Emporio : return true;
```

Fungsi pengecekan kelayakan suatu kartu kini telah

diperbaiki dengan beberapa kondisi. Berikut akan dijelaskan perubahan-perubahan tersebut. Untuk kartu :

- *Beer*, jika *life-points* penuh, maka tidak ada gunanya menggunakan Beer. Untuk kartu *Weapon,* akan ada pengecekan apakah kartu senjata volcanic atau bukan (diutamakan yang Volcanic terlebih dahulu).
- *Saloon* ditambah pengecekan tergantung karakter yang dimiliki, karena *Outlaw* biasanya sering melukai musuh, maka jika ada Saloon (efek seperti yang telah dijelaskan sebelumnya) biasanya kartu ini diabaikan.
- *Mustang* dan *Appaloosa* tergantung pada apakah kartu tersebut sudah ada di *field* pemain.
- *Dynamite* lebih digunakan jika *life-point* yang dimiliki hanya kurang dari sama dengan 2 (sekaligus bunuh diri).
- *Bang!, Cat Balou,* dan *Panico* digunakan ketika Bang diperbolehkan dan akan dicari target untuk ditembak. Pada proses `searchTarget()`, akan dicari musuh yang meungkinkan sebagai orang yang ditembak. `SearchTarget()` berdasarkan pada aksi-aksi yang dilakukan oleh pemain lain, bisa saja disimpan dalam bentuk suatu struktur data, misalnya -1 jika user tersebut merugikan, + jika sebaliknya.
- *Indiana* dan Gatling digunakan tergantung peran yang dimilikinya dan jika Sheriff memiliki darah kurang dari 3 untuk Deputy.

Dari analisis tadi, algoritma Greedy sederhana hanya menghasilkan suatu Bot AI yang kurang pintar. Dengan menambahkan informasi tambahan pada algoritma pengecekan kelayakan suatu kartu, kecerdasan Bot AI dapat terus dikembangkan.

Penambahan pengetahuan ini dapat dikembangkan sedemikian rupa, seperti menambahkan pengetahuan tentang karakter-karakter yang memiliki kemampuan unik yang spesial.

V. KESIMPULAN

Algoritma Greedy pada pemilihan kartu yang akan dikeluarkan saat giliran (*turn*) permainan Bang! merupakan algoritma yang sederhana. Algoritma ini dapat menjadi dasar dari pengembangan algoritma selanjutnya dengan menambahkan beberapa pengetahuan strategi dari manusia dengan pengondisian yang dapat diperkirakan oleh manusia.

Algoritma Greedy sederhana ini hanya menghasilkan Bot AI dengan tingkat kecerdasan yang rendah sehingga dirasakan. Dengan menambahkan pengetahuan-pengetahuan tersebut Bot AI dapat menjadi Bot berkemampuan menyerupai manusia untuk menambah keseruan dalam bermain permainan Bang! ini.

REFERENSI

[1] R. Munir,Strategi Algoritma, Teknik Informatika, Bandung, 2009.

[2] http://boardgamegeek.com/boardgame/3955/bang
Tanggal akses: 21 Desember 2012, pukul 00:00

[3] http://www.davincigames.it/download/Bang!_2nd_ed_rules.zip
Tanggal akses: 21 Desember 2012, pukul 00:00

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, 15 Desember 2010

ttd

Nugroho Satrijandi
13510032