PENGGUNAAN ALGORITMA *GREEDY* DALAM MENYELESAIKAN PERMAINAN *CHECKERS*

Tami Utiwi Handayani
(13506059)

Program Studi Teknik Informatika,Sekolah Teknik Elektro dan Informatika,Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
e-mail: if16059@students.if.itb.ac.id

ABSTRAK

Ada beberapa algoritma yang diimplementasikan dalam permainan yang tergolong *board games*. Salah satu algoritma yang diimplementasikan adalah algoritma *greedy* dan salah satu permainan yang termasuk ke dalam *board games* adalah permainan *checkers*.
Algoritma *greedy* adalah algoritma yang populer digunakan untuk pemecahan masalah optimasi. Algoritma ini sederhana dan lempang. Algoritma ini mengambil solusi optimum yang terdekat dengan posisinya sekarang. Algoritma *greedy* mengambil pilihan terbaik yang diperolehnya. Permainan *checkers* merupakan permainan yang tergolong *board games* yang cukup terkenal. Permainan *checkers* memanfaatkan koin-koin sebagai subyek permainan dan papan sebagai alas permainannya.
Makalah ini membahas tentang penggunaan algoritma *greedy* dalam permainan *checkers*. Makalah ini mencoba menyelesaikan permainan *checkers* menggunakan algoritma *greedy*. Pengunaan algoritma *greedy* cukup mangkus dalam menyelesaikan permainan *checkers* ini. Algoritma ini memungkinkan koin-koin *checkers* bergerak menuju solusi optimum. Solusi optimum yang dimaksud adalah dimana koin-koin *checkers* dapat mengambil sebanyak-banyaknya koin *checkers* lawan.

Kata kunci: Algoritma *greedy*, *checkers*, dan *board games*.

1. PENDAHULUAN

Bermain merupakan kebutuhan manusia. Bermain dapat memenuhi kebutuhan otak kiri manusia. Bermain juga dapat menyeimbangkan antara kerja otak kiri dan otak kanan manusia. Ada banyak jenis permaina yang sering dimainkan. Salah satu yang paling sering adalah permaian jenis *board games*. *Board games* adalah jenis permainan yang memanfaatkan papan sebagai lata bantu permainannya.

Dengan perkembangan teknologi saat ini, permainan yang biasa hanya dapat dimainkan secara nyata, dapat dimainkan secara virtual. Ada banyak jenis dan variasinya. Salah satunya adalah permainan komputer.

Pada perkembangannya, permainan jenis *board games* juga sudah dikembangkan secara virtual dengan komputer. Dengan keadaan seperti ini, permainan berjenis *board games* dimungkinkan dapat dimainkan hanya dengan satu orang pemain dan komputer yang telah dirancang seperti seorang manusia sebagai lawannya (atau sering disebut *artificial inteligince (AI)*).

Untuk membuat komputer agar dapat berperilaku seperti manusia, diperlukan sebuah algoritma. Algoritma ini ditujukan agar keputusan yang diambil oleh komputer tersebut dapat menyelesaikan permainan ini. Algoritma yang akan dibahas dalam makalah ini untuk menyelesaikan permainan ini adalah algoritma *greedy*.

Algoritma *greedy* adalah algoritma yang digunakan untuk masalah optimasi. Algoritma ini diharapkan mampu untuk menyelesaikan masalah penyelesaian permainan ini. Permainan yang akan dibahas untuk diselesaikan oleh algoritma *greedy* dalam makalah ini adalah salah satu dari permainan jeni *board games*, yaitu *checkers*.

2. DASAR TEORI

2.1 Algoritma *Greedy*

Algoritma *greedy* merupakan algoritma yang populer untuk memecahkan masalaha optimasi. Algoritma ini sederhana dan lempang. Prinsip dari algoritma *greedy* adalah "*take what can you get now*!".

Algoritma *greedy* membentuk solusi langkah per langkah. Pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan dan keputusan yang sudah diambil tidak bisa diubah lagi dan diulang.

Algoritma *greedy* menggunakan pendekatan membuat pilihan yang terlihat seperti solusi terbaik dengan membuat pilihan solusi optimum lokal. Dari solusi itu,

untuk setiap langkah yang membawa solusi optimum, diharapkan mampu menjadi solusi optimum global.

Persoalan optimasi dalam algoritma *greedy* disusun oleh elemen-elemen sebagai berikut:

1. Himpunan kandidat, C.
   Himpunan ini berisi dari himpunan-himpunan pembentuk solusi. Pada tiap langkah, satu buah kandidat diambil dari himpunannya.
2. Himpunan solusi,S.
   Himpunan ini berisi kandidat yang terpilih sebagai solusi dari persoalan yang diangkat
3. Fungsi seleksi.
   Fungsi yang ada pada setiap langkah dalam memilih kandidat yang paling memungkinkan mencapai solusi optimal.
4. Fungsi kelayakan.
   Fungsi yang memeriksa apakah kandidat yang dipilih dapat memberikan solusi yang layak, yakni kandidat bersama-sama dengan himpunan solusi yang terbentuk tidak melanggar kendala yang ada.
5. Fungsi obyektif.
   Fungsi yang mengoptimumkan (memaksimalkan atau meminimumkan) nilai solusi.

Namun algoritma *greedy* belum tentu menghasilkan solusi yang optimum global. Hal ini dikarenakan algoritma *greedy* tidak beroprasi secara menyeluruh terhadap semua alternatif solusi yang ada dan pemilihan fungsi seleksi yang biasanya didasarkan pada fungsi obyektif.

Berikut *pseudo-code* dari algoritma *greedy*:

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

Kamus
x : kandidat
S : himpunan_kandidat

Algoritma
S ← {}
while (not SOLUSI (S) and (C ≠ {}) )
do
x ← SELEKSI(C)
C ← C - {x}
      if LAYAK(S ∪{x})  then
      S ← S ∪{x}
      endif
 endwhile

if SOLUSI(S) then
      return S
endif
```

2.2 Permainan *Checkers*

Permainan *checkers* (draughts) merupakan salah satu jenis permaian *board games*. Dimana jenis permainan ini menggunakan alat bantu papan sebagai alas dan arena bermainnya.

Permainan ini dimainkan oleh 2 orang pemain. Satu pemain memainkan koin berwarna terang, dan lainnya memainkan koin berwarna gelap. Pemain dengan koin berwarna gelap memulai langkah awal terlebih dahulu.

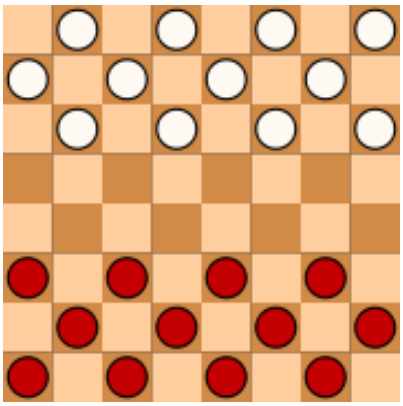
Beberapa komponen dalam permainan *checkers*:

1. Pemain.
   Permainan ini dimainkan oleh dua orang. Berperan sebagai lawan satu sama lain.
2. Papan.
   Papan yang digunakan dalam permainan ini mirip dengan papan catur. Terdiri dari 8 x 8 kotak-kotak kecil warna terang dan gelap.
3. Koin.
   Koin yang digunakan berbentuk silinder datar seperti koin pada umumnya. Koin yang digunakan pada permainan ini berwarna terang dan gelap. Pada awalnya, warna terang direpresentasikan dengan warna putih dan warna gelap direpresentasikan dengan warna merah. Tapi pada perkembangannya,banyak warna yang digunakan untuk menggantikan warna-warna tersebut.
4. Posisi awal.
   Masing-masing pemain mempunyai 12 koin sewarna yang diletakkan pada 3 baris pertama pada bidang yang terdekat dengan pemain dan diletakkan pada bidang yang berwarna gelap. Pemain yang memainkan koin berwarna gelap, memulai langkah awal terlebih dahulu.
5. Cara bergerak.
   Koin bergerak secara diagonal mengikuti warna bidang yang gelap, satu di tiap langkahnya. Cara lain adalah dengan melangkahi satu buah koin lawan. Keadaan itu mungkin dapat dilakukan jika pada diagonal setelah koin lawan, merupakan bidang kosong. Jika langkah kedua itu terjadi, koin lawan yang dilangkahi mati, dan harus keluar dari bidang permainan. Koin dengan pangkat "biasa" hanya dapat bergerak maju. Namun, koin dengan pangkat "raja", dapat bergerak maju maupun mundur.
6. Raja.
   Raja merupakan sebutan untuk koin yang mendapat perlakuan istimewa. Koin ini adalah koin yang bertahan dan berhasil masuk ke ujung baris bidang lawan. Koin ini didefinisikan sebagai tumpukan dari dua koin biasa. Keistimewaan koin ini adalah dapat bergerak baik maju maupun mundur.

7. Permainan selesai.
   Permainan selesai ditandai oleh habisnya koin lawan pada bidang permainan atau koin sudah tidak dapat bergerak kemanapun.

3. PENERAPAN ALGORITMA *GREEDY* PADA *CHECKERS*

Permainan *checkers* terdiri dari 24 buah koin yang digerakan secara bergantian sesuai warna koin. Terdapat 5 x 10^{20} kemungkinan untuk meletakkan koin-koin tersebut pada bidang permainan. Pada posisi awal, terdapat 14 kemungkinan langkah awal (7 untuk tiap warnanya).



Gambar 1. Posisi awal permainan *checkers*

Susunan koin di*assign* sebagai *array of* koin. Dengan algoritma *greedy*, pada langkah awal, koin yang dicek paling awal keberadaanya, dan ditemui ada jalan untuk melangkah dijalankan terlebih dahulu.

Gerakan selanjutnya ditentukan oleh posisi koin-koin, baik koin lawan maupun koin sendiri. Jika pada diagonal-diagonalnya (baik diagonal kiri maupun diagonal kanan) tidak terdapat jalan, maka periksa koin pada posisi lain. Jika pada diagonal-diagonalnya (baik diagonal kiri maupun diagonal kanan) terdapat jalan, maka periksa posisi koin-koin "mungkin jalan" lain yang memungkinkan untuk jalan. Jika ada posisi koin "mungkin jalan" yang dapat melangkahi koin lawan, jalan itu yang diambil oleh algoritma ini. Kondisi koin "mungkin jalan" yang dimaksud adalah kondisi dimana diagonal-diagonal koin kosong atau diagonal-diagonal koin terdapat koin lawan, tetapi setelahnya merupakan bidang kosong.

Algoritma *greedy* yang dibahas dalam makalah ini adalah *greedy by* jalan memakan koin lawan terdekat. Koin akan membuat prioritas berjalan. Jika koin menemukan pertama kali jalan yang dapat memakan koin lawan, maka jalan itu yang akan diambilnya. Jika tidak, ia akan bergerak saat menemukan kondisi "mungkin jalan" yang pertama.

Penentuan pengambilan keputusan didasarkan pada algoritma *greedy*. Algoritma *greedy* akan memeriksa koin "mungkin jalan". Koin akan mendapat status koin "mungkin jalan 1" atau koin "mungkin jalan 2". Status yang pertama merupakan status koin hanya berjalan biasa. Status kedua menunjukan bahwa koin berjalan dengan memakan koin lawannya.

Berikut *pseudo-code* algoritma yang digunakan:

```
procedure greedy (input C:
himpunan_jalan, output S:
himpunan_jalan)
{menggerakkan koin yang mempunyai
jalan untuk mengambil koin lawan}

Kamus
S : himpunan_jalan
k : koin
koin : 1

algoritma
while ((posisi koin ≠ posisi
terakhir) and C ≠ {}) do
D ← cekKondisi(k)
if (D = kondisi buntu) then
      k ← k+1
else
      if (Status(k) = kondisi
      "mungkin jalan 2") then
            posisi(k) ← posisi(k[(i
      +2),(j+2)])
      else
            posisi(k) ← posisi(k[(i
      +1),(j+1)])
      endif
endif
endwhile
```

Pada algoritma tersebut, terdapat fungsi-fungsi pendukung. Fungsi-fungsi itu memungkinkan koin diberi status terlebih dahulu baru kemudian dieksekusi pada program utama.

4. ANALISIS

Algoritma *greedy* akan selalu mengambil jalan yang paling cepat ia temukan terdapat koin yang dapat diambil. Koin-koin yang bergerak, tidak mementingkan posisi selanjutnya ia berada. Maka dari itu, tidak menutup kemungkinan kita akan kalah dalam permainan ini. Algoritma yang dengan sempurna menyelesaikan permainan *checkers* adalah algoritma *best first search* dan *depth first search*.

5. KESIMPULAN

1. Algoritma *greedy* merupakan algoritma yang dapat digunakan untuk menyelesaikan permainan *checkers*.
2. Algoritma *greedy* yang dibahas memberikan solusi optimum untuk mengambil koin lawan sebanyak-banyaknya.
3. Algoritma *greedy* yang dibahas pada makalah ini tidak memberikan solusi untuk menang.
4. Algoritma *greedy* mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi kedepannya.
5. Koin pada *checkers*, dengan menggunakan algoritma *greedy* ini akan bergerak sesuai kondisi "mungkin jalan 2" atau kondisi "mungkin jalan 1".
6. Algoritma yang dapat menyelesaikan permainan *checkers* dengan sempurna menggunakan algoritma *best first search* dan *depth first search*.
7. Permainan yang menggunakan komputer yang dirancang untuk berperilaku seperti manusia disebut juga *artificial inteligince(AI)*.

REFERENSI

[1] Munir,Rinaldi, "Diktat Kuliah IF2251 Strategi Algoritmik", Program Studi Teknik Informatika STEI ITB, 2006.

[2] Chinook, world Man-machine *checkers* champion.2007. http://www.cs.ualberta.ca/~chinook Tanggal akses : 19 Mei 2008, pukul 14.35

[3] Wikipedia.2008. http://en.wikipedia.org/wiki/*Checkers* tanggal akses : 19 Mei 2008, pukul 20.00