PENGGUNAAN ALGORITMA GREEDY DALAM INTELEGENSIA BUATAN PERMAINAN REVERSI

Aqsath Rasyid Naradhipa

Jurusan Teknik Informatika
Sekolah Teknik Elektro Informatika
Institut Teknologi Bandung
Jalan Ganeca 10, Bandung
e-mail : aqsath@gmail.com

ABSTRAK

Algoritma Greedy adalah algoritma yang memegang prinsip "*take what you can get now*!" sehingga algoritma ini akan mengambil apa yang bisa diambil sekarang dan sering dipakai untuk memcahkan persoalan optimasi. Namun, disini saya menggunakannya untuk pembuatan intelegensia buatan pada permainan reversi. Karena pada dasarnya permainan reversi juga memegang prinsip yang hampir sama, yaitu mengambil sebanyak mungkin biji lawan yang mungkin. Sehingga intelegensia buatan yang dihasilkan akan menjadi "lawan" yang cukup sulit bagi penantangnya karena algoritma yang dipakai akan mencari solusi optimum dan mengambil biji dari lawannya sebanyak mungkin. Intelegensia buatan dalam reversi mempunyai syarat – syarat agar penggunaan algoritma greedy dapat berjalan dengan baik, yaitu properti pilihan greedy dan substruktur yang optimal. Sehingga penggunaan algoritma greedy di dalam perancangan intelegensia buatan permainan reversi akan sangat baik.

Kata kunci, Algoritma Greedy, Intelegensia buatan, Reversi

1. PENDAHULUAN

Reversi adalah permainan yang sangat mengandalkan jumlah terbanyak dalam permainannya. Jadi, siapa pemain yang mendapatkan biji terbanyak sampai permainan selesai maka dialah yang menang. Dan jumlah terbanyak ini adalah konsep yang sangat dipegang oleh algoritma greedy. Jadi, penggunaan algoritma greedy di dalam pembuatan intelegensia buatan pada permainan reversi akan mendapatkan solusi yang optimum sehingga intelegensia buatan yang dihasilkan pun akan menjadi lawan yang tangguh dalam permainannya.

2. ALGORITMA GREEDY

Greedy adalah algoritma yang memecahkan masalah dengan metaheuristik dengan membuat solusi optimum lokal di tiap langkah dan berharap menemukan solusi optimum global. Metahuristik berasal dari bahasa yunani yaitu meta dan heuristik. Meta berarti "di atas" dan heuristik berasal dari kata heuriskein yang berarti "untuk mencari", jadi metaheuristik bisa berarti untuk mencari ke tingkatan yang lebih tinggi.

Biasanya, algoritma Greedy mempunya lima pilar, yaitu :

1. Himpunan kandidat
   Himpunan ini berisi elemen – elemen pembentuk solusi.

2. Himpunan Solusi
   Berisi kandidat – kandidat yang terpilih sebagain solusi persoalan. Dengan kata lain, himpunan solusi adalah himpunan bagian dari himpunan kandidat.

3. Fungsi Seleksi
   Fungsi yang pada setiap langkah memilih kandidat yang paling memungkinkan mencapai solusi optimal. Kandidat yang sudah dipilih pada suatu langkah tidak pernah dipertimbangkan lagi pada langkah selanjutnya. Biasanya setiap kandidat, x, ditunjuk sebuah nilai numerik, dan gungsi seleksi memilih x yang mempunyai nilai terbesar atau memilih x yang mempunyai nilai terkecil.

4. Fungsi Kelayakan
   Fungsi yang memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yakni kandidat tersebut bersama – sama dengan himpunan solusi yang yang sudah terbentuk tidak melanggar kendala (*constraints*) yang ada. Kandidat yang layak dimasukkan ke dalam himpunan solusi,

sedangkan kandidat yang tidak layak dibuang dan tidak pernah dipertimbangkan lagi.

5. Fungsi obyektif, yaitu fungsi yang memaksimumkan atau meminimumkan nilai solusi (misalnya panjang lintasan, keuntungan, dan lain – lain).

Algoritma Greedy memproduksi solusi yang baik untuk beberapa masalah matematika, tetapi tidak untuk yang lain. Masalah yang dapat diselesaikan dengan algoritma greedy mempunyai dua properti, yaitu :

1. Properti Pilihan Greedy
   Kita dapat membuat pilihan apapun yang terlihat terbaik di saat itu dan menyelesaikan bagian dari masalah yang muncul selanjutnya. Dia meng-iterasi membuat satu pilihan greedy setelah yang lainnya, mereduksi satu masalah menjadi lebih kecil (rekursif). Dengan kata lain, greedy tidak pernah memikirkan lagi sesuatu yang sudah dipilihnya sebelumnya. Inilah perbedaan yang terbesar dari pemrograman dinamis yang melakukannya secara *exhaustive* dan menjamin untuk mencara solusi. Setelah melewati satu tahap, pemrograman dinamis akan membuat keputusan yang berdasarkan kepada semua keputusan yang sudah dibuat di tahap sebelumnya, dan memungkinkan untuk kembali ke tahap selanjutnya untuk mencari solusinya.

2. Sub-Struktur yang Optimal
   Di dalam ilmu komputer, sebuah masalah bisa dikatakan mempunyai sub-struktur yang optimal jika solusi optimal bisa dibangun secara efisien dari solusi optimal di tiap sub-masalahnya.

Untuk beberapa masalah, algoritma greedy bisa membuat solusi kemungkinan yang malah lebih buruk. Algoritma greedy biasanya (tetapi tidak selalu) gagal mencari solusi optimal global, karena algoritma greedy tidak melakukan operasi secara *exhaustive* kepada semua data. Algoritma greedy dapat tidak mementingkan apakah mendapatkan solusi terbaik di akhirnya. Walau begitu, algoritma greedy sangat berguna karena algoritma greedy cepat dalam berpikirdan sering memberikan perkiraan yang bagus dalam nilai optimum.

Jika algoritma greedy bisa membuktikan nilai optimum global untuk masalah yang telah diberikan., biasanya itu bisa menjadi pilihan karena algoritma greedy lebih cepat dari metode optimisasi lainnya seperti pemrograman dinamis.

3. INTELEGENSIA BUATAN

Kebanyakan buku – buku teks tentang intelegensia buatan mendefinisikan intelegensia buatan adalah sebuah studi dan pernacangan agen intelegensia. Agen intelegensia adalah sebuah sistem yang mengenali lingkungannya dan mengambil aksi yang akan memaksimalkan kemungkinan keberhasilannya. Intelegensia buatan dapat dilihat sebagai bentuk realisasi dari abstrak agen intelegensia yang memperlihatkan fungsi esensi dari intelegensia. John McCarthy mendefinisikannya “ilmu yang membuat mesin menjadi pintar”.

Para peneliti berharap mesin akan dapat melakukan pengetahuan, perencanaan, pembelajaran, komunikasi, persepsi dan kemampuan untuk menggerakkan dan memanipulasi objek. Intelegensia general (atau Intelegensia Buatan yang kuat) masih belom ditemukan dan menjadi tujuan jangka panjang dari pengembangan intelegensia buatan.

Pengembangan intelegensia buatan menggunakan alat dan sarana dari banyak bidang, termasuk ilmu komputer, psikologi, filosofi, *neuroscience,* ilmu kognitif, bahasa, ontologi, penelitian operasi, ekonomi, teori kendali, probabilitas, optimisasi, dan logika. Pengembangan intelegensia buatan juga termasuk robotik, sistem kendali, penjadwalan, pengumpulan data, logistik, pengenalan suara, pengenalan bentuk muka dan lainnya. Orang lain juga ada yang menyebur intelegensia buatan dengan sebutan lain seperti intelegensia komputer, intelegensia sintetis, sistem intelijen, atau komputasi rasional.

Manusia telah mengimajinasikan sebab akibat dari mesin yang berfikir atau intelegensia buatan yang sangat detail.Intelegensia buatan pertama kali muncul mitologi yunani seperti Talos dari Crete, robot emas dari Hephaestus dan Galatea Pygmalion. Robot yang menyerupai manusia yang paling pertama diketahui adalah patung yang disembah di Mesir dan Yunani, yang dipercaya telah dikarunai sebuah bakat oleh pembuat patungnya. Di jaman Medieval, para ilmuwan, seperti Paracelcus telah mengatakan membuat intelegensia buatan.

Penelitian tentang intelegensia buatan pertama kali mengembangkan algoritma yang menyerupai proses dari permasalahannya, langkah – langkah yang dilakukan manusia untuk menyelesaikan puzzle, bermain permainan papan, atau membuat deduksi logika. Di akhir tahun 80-an dan 90-an, penelitian intelegensia buatan juga mengembangkan metode yang mempunyai tingkat keberhasilan tinggi untuk menyelesaikan suatu

masalah dengan informasi yang tidak lengkap, konsep pengembangannya berasal dari probabilitas dan ekonomi.

Untuk masalah yang lebih sulit, kebanyakan dari algoritma ini dapat meminta bahan komputasi, pengalaman yang lebih akan menjadi ledakan kombinatorial : jumlah memori atau waktu yang dibutuhkan komputer akan menjadi angka astronomi jika masalahnya menjadi diatas ukuran yang biasanya. Pencarian untuk penyelesaian masalah yang efisien adalah prioritas yang paling tinggi dalam penelitian intelegensia buatan.

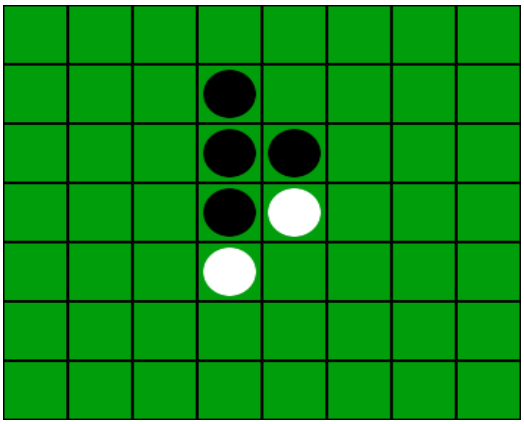
Bagaimanapun juga, pemikiran sadar manusia lebih efisien ketika berhadapan dengan masalah abstrak yang sulit. Peneliti kognitif telah meunjukkan bahwa manusia lebih banyak menyelesaikan masalahnya dengan pemikiran di bawah kesadaran mereka, daripada dengan kesadaran mereka. Deduksi langkah – langkah pada penelitian intelegensia buatan yang sebelumnya bisa dijadikan sebuah model. Peneliti kognitif berpendapat bahwa kemampuan sensorimotor bawah sadar sangat penting untuk kemampuan menyelesaikan masalah. Metode sub-simbolik seperti intelegensia komputasi dan Intelegensia buatan yang khusus diharapkan dapat memodelkan kemampuan instingnya. Masalah dari penyelesain masalah bawah sadar ini, membentuk bagian dari pemikiran yang masuk akal yang sangat susah untuk diselesaikan.

4. REVERSI

Reversi (juga biasa disebut Othello) adalah sebuah permainan strategi yang dimainkan oleh dua orang di papan yang mempunya garis 8 X 8 dan mempunyai biji dengan dua buah warna. Bijinya biasanya berbentuk seperti koin, tetapi dengan gambar hitam dan putih yang merepresentasikan tiap pemain. Tujuan dari permainan ini adalah membuat biji anda memiliki bagian paling banyak (mayoritas) dari papan yang dimainkan pada akhir permainan.

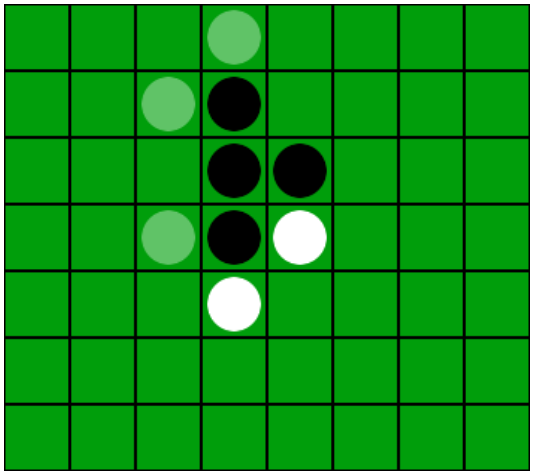
5. APLIKASI ALGORITMA GREEDY DALAM PEMBUATAN INTELEGENSIA BUATAN

Intelegensia buatan akan dirancang untuk selalu mengambil biji sebanyak mungkin pada saat gilirannya bergerak sehingga diharapkan dengan mempunyai selalu mengambil biji terbanyak di tiap giliran maka akan mempunyai biji terbanyak pada akhir permainan.



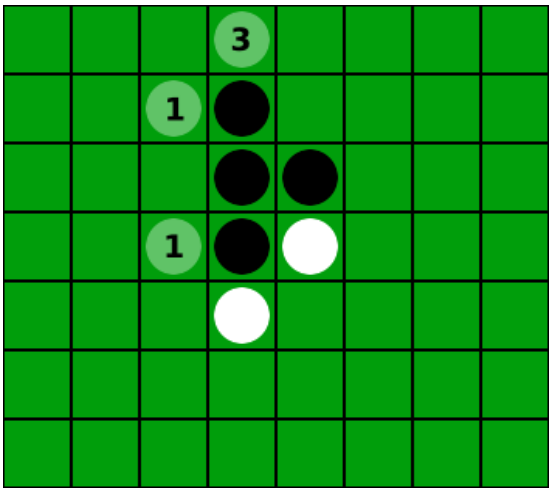
Ketika AI mendapat giliran untuk melakukan gerakan maka dia akan memindai papan dan melihat kotak mana yang memungkinkan untuk bergerak.

```
for  (i := 1 to 8) do begin
 for (j := 1 to 8) do begin
  if (BisaJalan(papan[i][j])) then
begin
   listKotak.add(i,j)
  endif
 endfor
endfor
```



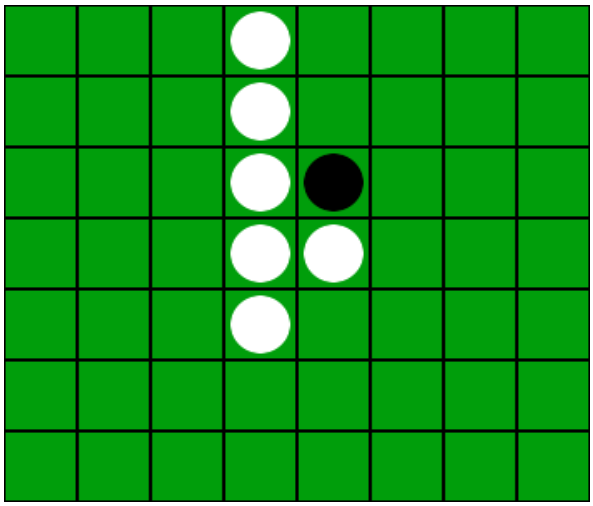
Setelah pemindaian selesai maka akan dihitung berapa biji lawan yang akan terganti bila dia melakukan gerakan itu.

```
for ((x = list.first) != null) do
begin
 HitungBiji(x)
  x = x.next
endfor
```



Lalu akan dihitung nilai maksimumnya. Jika memang nilai itu maksimum maka AI akan menggerakannya ke kotak itu.

```
AI.move(i,j)
```



REFERENSI

[1] Munir, Rinaldi. Diktat Kuliah Strategi Algoritmik. 2007. Bandung : Program Studi Teknik Informatika ITB.

[2] Cormen, dkk. Introduction to Algorithms Chapter 16 “Greedy Algorithms”. 2001.

[3] Searle, John. GPS : A Program that Simulates Human Thought. 1980. McGraw-Hill.