BEBERAPA IMPLEMENTASI ALGORITMA GREEDY DALAM PERMAINAN CONGKLAK

Anis Kamilah Hayati (13505075)

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Ganeca 10 Bandung

e-mail: if15075@students.if.itb.ac.id

ABSTRAK

Algoritma *Greedy* memiliki banyak sekali contoh implementasi dalam masalah optimasi. Hal tersebut terutama karena sifatnya yang khas, dengan prinsip "*take what you can get now*!" [3] .

Adapun Congklak adalah sejenis permainan tradisional yang cukup dikenal di berbagai daerah. Congklak dikenal dengan berbagai macam nama di seluruh Indonesia. Di Malaysia permainan ini lebih dikenal dengan nama *congkak* dan istilah ini juga dikenal di beberapa daerah di Sumatera dengan kebudayaan Melayu. Di Jawa, permainan ini lebih dikenal dengan nama Congklak, *dakon*, *dhakon* atau *dhakonan*. Selain itu di Lampung permainan ini lebih dikenal dengan nama *dentuman lamban* sedangkan di Sulawesi permainan ini lebih dikenal dengan nama *Mokaotan*, *Maggaleceng*, *Aggalacang* dan *Nogarata*. Dalam bahasa Inggris, permainan ini disebut *Mancala*[1]

Permainan ini bertujuan untuk mendapatkan sebanyak-banyaknya biji congklak (biasanya sejenis cangkang kerang lokan atau biji-bijian [4]). Sehingga kita dapat mengkategorikan permainan ini sebagai permainan optimasi.

Dalam makalah ini penulis mencoba mengimplementasikan algoritma *Greedy* untuk mencari beberapa solusi optimum dalam permainan Congklak.

Kata kunci: Congklak, *Greedy*.

1. PENDAHULUAN

1.1 Peraturan Permainan

Pada umumnya permainan Congklak dimainkan oleh dua orang pemain. Peralatan yang dibutuhkan adalah sebuah papan dengan 16 lubang yang terdiri dari 14 lubang kecil dan dua lubang besar (tujuh lubang kecil dan satu lubang besar untuk masing-masing pemain), serta 98 (14 x 7) biji congklak.

Setelah setiap lubang kecil diisi dengan tujuh biji, pemain dengan giliran pertama mengambil seluruh biji yang terdapat pada salah satu lubang kecil dan membagikan biji tersebut pada lubang-lubang (kecuali lubang besar milik lawan) dalam arah searah jarum jam, sampai habis. Jika biji habis di lubang yang berisi biji, maka pemain tersebut dapat terur bermain sampai ia mati sehingga pemain kedua mendapat giliran bermain.

Keadaan mati yaitu ketika biji habis di lubang yang kosong. Jika pemain mati di lubang miliknya, sementara pada lubang lawan yang berhadapan dengan lubang tempat ia mati terdapat biji, maka ia berhak mengambil seluruh biji yang terdapat pada lubang lawan yang berhadapan itu, dalam makalah ini keadaan tersebut dinamakan menembak. Tapi jika pemain mati di lubang milik lawan, pemain tersebut tidak berhak melakukan apapun.

Permainan ini akan terus berlanjut sampai tak ada lagi biji pada lubang kecil salah satu atau kedua pemain. Pemain yang dikatakan memenangkan permainan adalah pemain yang berhasil memasukkan biji paling banyak ke dalam lubang besar miliknya.[2]

1.2 Prinsip Umum Algoritma *Greedy*

Algoritma *Greedy* adalah algoritma yang memecahkan masalah langkah per langkah, pada setiap langkah:

1. mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatika konsekuensi ke depan
2. berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global.

Skema umum Algoritma *Greedy*:

1. Himpunan kandidat, himpunan ini berisi seluruh elemen pembentuk solusi
2. Himpunan solusi, berisi kandidat-kandidat yang terpilih sebagai solusi persoalan
3. Fungsi seleksi, fungsi yang pada setiap langkah memilih kandidat yang paling memungkinkan mencapai solusi optimal
4. Fungsi kelayakan, fungsi yang memeriksa apakah suatu kandidat yang terpilih dapat memberikan solusi

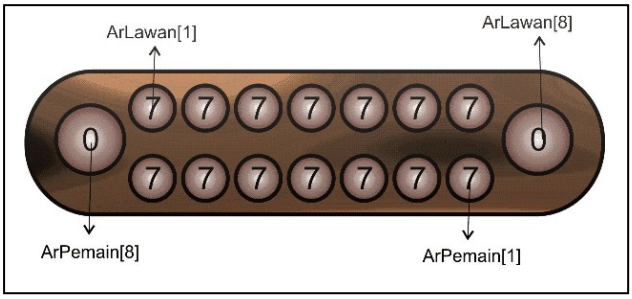
yang layak, yaitu kandidat bersama-sama dengan himpunan solusi yang sudah terbentuk tidak melanggar kendala yang ada.

5. Fungsi objektif, fungsi yang memaksimumkan atau meminimumkan nilai solusi. [3]

1.3 Representasi Masalah

Dalam makalah ini, penulis merepresentasikan beberapa hal, yaitu:

1. Dua buah larik (*array*) yang masing-masing larik merepresentasikan satu lubang besar dan tujuh lubang kecil milik pemain. Dalam makalah ini larik tersebut penulis namai ArPemain dan ArLawan. Indeks ArLawan dimulai dari kiri ke kanan sementara indeks ArPemain dimulai dari kanan ke kiri. Sehingga indeks ke-8 selalu menunjukkan lubang besar.
2. Angka merepresentasikan jumlah biji yang terdapat pada suatu lubang



Gambar 1. Representasi Masalah

2. PEMBAHASAN

2.1 Ruang Lingkup Pembahasan

Pada makalah ini, penulis membatasi pembahasan pada beberapa strategi untuk memenangkan permainan Congklak, strategi tersebut diantaranya:

1. Menentukan lubang yang akan dimainkan untuk menembak lubang lawan dan mendapatkan biji terbanyak dengan satu atau dua kali mengambil seluruh biji dari suatu lubang.
2. Menentukan lubang yang akan dimainkan agar pemain dapat bermain minimal dengan dua kali mengambil seluruh biji dari dua lubang.

3. PEMILIHAN LUBANG UNTUK OPTIMASI TEMBAKAN

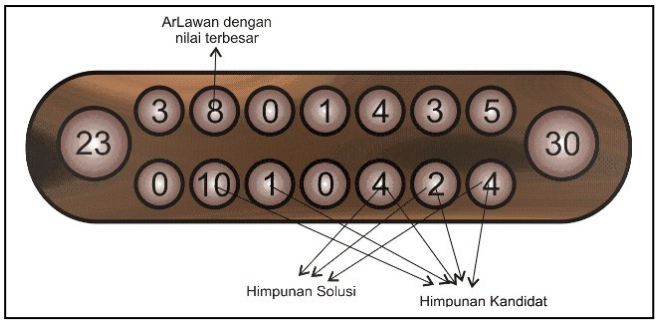
Untuk menjelaskan strategi ini, indeks ArPemain yang akan dimainkan dinamakan x, indeks ArPemain tempat menembak (lubang kosong) dinamakan y, dan indeks ArLawan yang akan ditembak (lubang lawan yang berhadapan dengan lubang kosong milik pemain) dinamakan z.

Untuk setiap ArLawan dengan indeks j yang berhadapan dengan ArPemain dengan indeks i, j dirumuskan sebagai berikut: j = 8 – i sehingga z = 8 – y

3.1 Strategi Untuk Persoalan x < y

Dalam persoalan ini, kita tidak mungkin melewati lubang lawan karena setiap kali melangkah, lubang-lubang yang terlewati akan diisi satu biji untuk setiap lubang. Dengan demikian lubang dengan indeks y yang menjadi sasaran kita ikut terisi.

Dengan menggunakan algoritma *Greedy*, kita menentukan larik bernilai nol ArPemain[j] sedemikian sehingga ArLawan[8-j] bernilai paling besar.



Gambar 2. Contoh Persoalan x < y

Dengan menggunakan skema umum Algoritma *Greedy*, kita dapat menentukan elemen-elemen persoalan ini sebagai berikut:

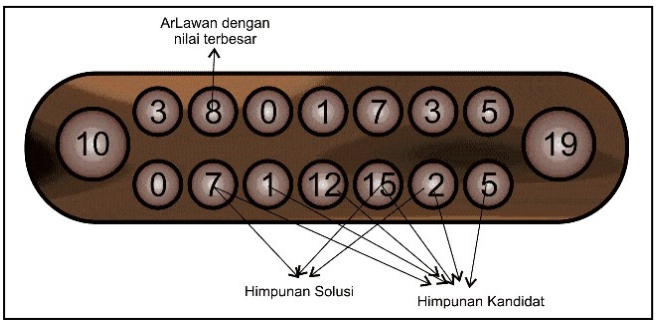
1. Himpunan kandidat (C) adalah seluruh indeks (i) ArPemain dimana ArPemain[i] tidak bernilai nol.
   $C=\{i \mid 0< i < 8, ArPemain[i] \neq 0\}$
2. Himpunan solusi (S) adalah indeks ArPemain sedemikian sehingga untuk ArPemain[i] indeks i<j dimana indeks j menunjukkan ArPemain yang bernilai nol, ArLawan yang berhadapan dengan ArPemain yang bernilai nol, tidak bernilai nol
   $S=\{i \mid i < j, ArPemain[j] = 0, ArLawan[8-j] \neq 0\}$
3. S disebut layak jika isi ArPemain[i] sama dengan jumlah langkah menuju ArPemain[j] yang bernilai nol. Atau terdapat ArPemain[k] dimana isi ArPemain[i] ditambah isi ArPemain[k] sama dengan jumlah langkah menuju ArPemain[j]
   ArPemain[i] = j – i atau
   ArPemain[i] + ArPemain[k] + 1= j – i
   (nilai 1 didapat dari satu biji yang diberikan ArPemain[i] untuk ArPemain[k])
4. Fungsi objektif adalah memaksimalkan isi ArPemain[8]. Dalam masalah ini, isi ArPemain[8] setelah pemain menembak adalah isi ArPemain[8] sebelum menembak ditambah isi ArLawan[j] yang ditembak.
   ArPemain[8] = ArPemain[8] + ArLawan[8-j]

5. Fungsi seleksi adalah memilih ArPemain[i] sedemikian sehingga isi ArPemain[i] atau isi ArPemain[i] ditambah isi ArPemain[k] samadengan jumlah langkah menuju ArPemain[j] dengan ArLawan[8-j] yang bernilai paling besar.

3.2 Strategi Untuk Persoalan x = y

Untuk persoalan ini, lubang yang dipilih akan menjadi kosong, terjadi satu kali putaran sampai kembali pada lubang yang sama. Untuk melakukan hal ini dibutuhkan 15 langkah, yaitu delapan langkah pada lubang milik pemain dan tujuh langkah pada lubang milik lawan.

Dengan menggunakan algoritma *Greedy*, kita menentukan larik bernilai nol (tempat penembakan) ArPemain[j] sedemikian sehingga ArLawan[8-j] bernilai paling besar.



Gambar 3. Contoh Persoalan x = y

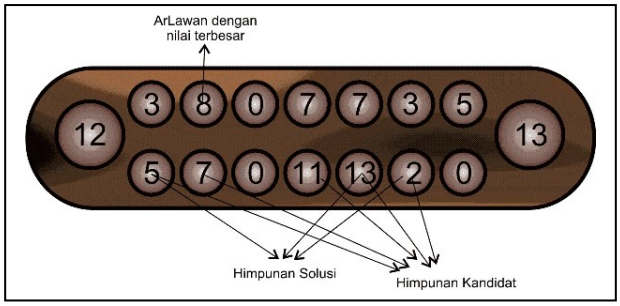
Dengan menggunakan skema umum Algoritma *Greedy*, kita dapat menentukan elemen-elemen persoalan ini sebagai berikut:

1. Himpunan kandidat (C) adalah seluruh indeks (i) ArPemain dimana ArPemain[i] tidak bernilai nol.
   C={i | 0< i < 8, ArPemain[i] ≠ 0}
2. Himpunan solusi (S) adalah indeks (i) ArPemain sedemikian sehingga untuk ArPemain[i] terdapat ArLawan yang berhadapan dengan ArPemain[i] yang tidak bernilai nol
   S={i | ArLawan[8-i] ≠ 0}
3. S disebut layak jika isi ArPemain[i] sama dengan lima belas, yaitu jumlah langkah untuk kembali ke ArPemain[i] yang sudah bernilai nol. Atau terdapat ArPemain[k] atau ArLawan[n] sedemikian sehingga jumlah langkah tepat lima belas sampai kembali ke ArPemain[i].
   ArPemain[i] = 15 atau
   ArPemain[i] + ArPemain[k] + 1 = 15 atau
   ArPemain[i] + ArLawan[n] + 1 = 15
   k = ArPemain[i] + i
   n = ArPemain[i] – (8 - i)
   (nilai 1 didapat dari satu biji yang diberikan ArPemain[i] untuk ArPemain[k] atau ArLawan[n])
4. Fungsi objektif adalah memaksimalkan isi ArPemain[8]. Dalam masalah ini, isi ArPemain[8] setelah pemain menembak adalah isi ArPemain[8] sebelum menembak ditambah isi ArLawan[j] yang ditembak.
   ArPemain[8] = ArPemain[8] + ArLawan[8-j]
5. Fungsi seleksi adalah memilih ArPemain[i] dengan ArLawan[8-i] yang bernilai paling besar.

3.3 Strategi Untuk Persoalan x > y

Untuk persoalan ini, lubang yang dipilih memiliki indeks yang lebih kecil dibanding lubang kosong tempat penembakan (x > y). Sehingga untuk persoalan ini kita pasti melewati lubang milik lawan.

Dengan menggunakan algoritma *Greedy*, kita menentukan indeks larik tempat penembakan, ArPemain[j] sedemikian sehingga ArLawan[8-j] bernilai paling besar.



Gambar 4. Contoh Persoalan x > y

Dengan menggunakan skema umum Algoritma *Greedy*, kita dapat menentukan elemen-elemen persoalan ini sebagai berikut:

1. Himpunan kandidat (C) adalah seluruh indeks (i) ArPemain dimana ArPemain[i] tidak bernilai nol.
   C={i | 0< i < 8, ArPemain[i] ≠ 0}
2. Himpunan solusi (S) adalah indeks ArPemain sedemikian sehingga untuk ArPemain[i] indeks i>j dimana indeks j menunjukkan ArPemain yang bernilai nol, ArLawan yang berhadapan dengan ArPemain yang bernilai nol, tidak bernilai nol
   S={i | i > j, ArPemain[j] = 0, ArLawan[8-j] ≠ 0}
3. S disebut layak jika isi ArPemain[i] sama dengan jumlah langkah menuju ArPemain[j] yang bernilai nol. Atau terdapat ArPemain[k] atau ArLawan[n] dimana isi ArPemain[i] ditambah isi ArPemain[k] atau ArLawan[n] sama dengan jumlah langkah menuju ArPemain[j].
   ArPemain[i] = 15 – (i – j) atau
   ArPemain[i] + ArPemain[k] + 1= 15 – (i - j) atau
   ArPemain[i] + ArLawan[n] + 1= 15 – (i - j) atau
   k = ArPemain[i] + i
   n = ArPemain[i] – (8 - i)
   (nilai 1 didapat dari satu biji yang diberikan ArPemain[i] untuk ArPemain[k] atau ArLawan[n])

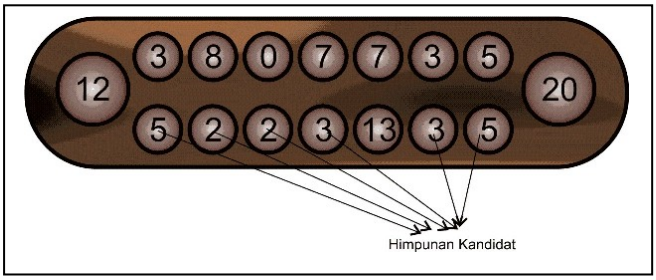
4. Fungsi objektif adalah memaksimalkan isi ArPemain[8]. Dalam masalah ini, isi ArPemain[8] setelah pemain menembak adalah isi ArPemain[8] sebelum menembak ditambah isi ArLawan[j] yang ditembak.
   ArPemain[8] = ArPemain[8] + ArLawan[8-j]
5. Fungsi seleksi adalah memilih ArPemain[i] sedemikian sehingga isi ArPemain[i] atau isi ArPemain[i] tambah ArPemain[k]/ ArLawan[n] sama dengan jumlah langkah menuju ArPemain[j] dengan ArLawan[8-j] yang bernilai paling besar.

4. MEMILIH LUBANG AGAR DAPAT BERMAIN LEBIH LAMA

Untuk menjelaskan strategi ini, indeks ArPemain yang akan dimainkan kita namakan x.

Dengan Algoritma *Greedy*, kita dapat menentukan indeks i pada ArPemain[i] untuk dimainkan dengan isi ArPemain[i] terbanyak.

Pada strategi ini, kita akan mencari cara untuk memasukkan biji terakhir pada lubang besar (ArPemain[8]) agar mendapat giliran untuk bermain lagi. Jika tidak ada, kita akan mencari lubang yang memiliki jumlah biji sehingga mengantarkan pada lubang besar.



Gambar 5. Contoh Persoalan Optimasi Langkah

Dengan menggunakan skema umum Algoritma *Greedy*, kita dapat menentukan elemen-elemen persoalan ini sebagai berikut:

1. Himpunan kandidat (C) adalah seluruh indeks (i) ArPemain dimana ArPemain[i] tidak bernilai nol.
   $C=\{i \mid 0< i < 8, ArPemain[i] \neq 0\}$
2. Himpunan solusi (S) adalah seluruh indeks (i) ArPemain sedemikian sehingga untuk ArPemain[i], jumlah setiap ArPemain[i] kurang dari atau sama dengan jumlah langkah menuju ArPemain[8].
   $S=\{i \mid \sum ArPemain[i] > 8 - i\}$
3. S disebut layak jika isi ArPemain[i] tidak sama dengan jumlah langkah menuju suatu larik ArPemain[k] atau ArLawan[n] yang bernilai nol.
   $ArPemain[i] \neq k - i$ atau
   $ArPemain[i] \neq n + (8 - i)$
4. Fungsi objektif adalah meminimalkan langkah menuju ArPemain[8]. Karena kita ingin memanfaatkan lubang dengan isi sejumlah langkah menuju ArPemain[8] yang terdekat sebelum ia diubah oleh langkah lain yang lebih jauh.
5. Fungsi seleksi adalah memilih ArPemain[i] terdekat dengan ArPemain[8], yaitu ArPemain dengan indeks terbesar.

Pada contoh di atas (pada gambar 5), kita dapatkan kemungkinan solusi:
S={6} dengan nilai ArPemain[2]=2 dan
S={2, 5} dengan nilai ArPemain[2]=3 dan ArPemain[5]=2.

5. KESIMPULAN

Kesimpulan yang dapat kita ambil dari makalah ini diantaranya:

1. Algoritma *Greedy* dapat digunakan pada persoalan optimasi, walau tidak selalu mendapatkan hasil optimum
2. Algoritma pencarian lubang yang akan dimainkan untuk mendapatkan ArPemain[8] terbesar hasil menembak, akan selalu didapatkan hasil optimum karena ArPemain[8] yang baru = ArPemain[8] yang lama + ArLawan[i] yang ditembak. Maka untuk setiap ArLawan[i] terbesar (yang termasuk himpunan solusi dan telah dinyatakan layak) merupakan hasil paling optimum untuk mendapatkan ArPemain[8] terbesar.
3. Algoritma pencarian lubang agar dapat bermain lebih lama, akan selalu menghasilkan langkah minimum. Karena untuk setiap ArPemain[i] yang paling dekat, akan terdapat indeks i paling besar yang akan mengakibatkan jumlah langkah (8 – i) paling kecil.
4. Kedua algoritma yang telah dijelaskan dalam makalah ini tidak dapat dibandingkan satu sama lainnya untuk mendapatkan solusi terbaik untuk memenangkan permainan, karena keduanya berbeda kasus (untuk menembak kita mencari lubang kosong, sementara untuk terus bermain kita mencari lubang yang tidak kosong).
5. Kedua algoritma yang telah dijelaskan dapat dikombinasikan untuk mendapatkan solusi yang lebih optimum. Misalnya algoritma mencari lubang kosong mendapat prioritas pertama sebelum algoritma mencari jalan menuju ArPemain[8].
6. Hasil kombinasi kedua algoritma belum tentu selalu menghasilkan kemenangan karena masih banyak komponen lain pada permainan ini yang belum diatasi, misalnya kondisi dimana tidak terdapat lubang dengan isi kurang dari langkah menuju ArPemain[8] tetapi lebih dari 15 untuk sampai pada lubang yang sama untuk menembak.

REFERENSI

[1] *Expat Web Site Association*, "*Congklak: Traditional Game of Indonesia*", http://www.expat.or.id/info/congklak.html. Diakses tanggal 21 Mei 2007 pukul 17.59

[2] *Expat Web Site Association*, "*Congklak: Instructions for Play*", http://www.expat.or.id/info/congklakinstructions.html . Diakses tanggal 21 Mei 2007 pukul 17.59

[3] Munir, Rinaldi, "Strategi Algoritmik", Teknik Informatika ITB, 2006.

[4] Wikipedia, "Congklak", http://id.wikipedia.org/wiki/Congklak. Diakses tanggal 21 Mei 2007 pukul 17.59