Implementasi Agoritma Program Dinamis dalam Permainan Tradisional Congklak

Fawwaz Muhammad 13511083
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
fawwazmuhammad@gmail.com

***Abstrak*—Congklak adalah salah satu permainan tradisional yang mulai jarang dimainkan. Dalam permainan congklak, setiap pemain memiliki tujuan untuk mengisi lubang rumahnya dengan biji congklak mengikuti aturan tertentu. Pemain yang memiliki biji congklak terbanyak di rumahnya yang menang. Permasalahan ini dapat ditransformasikan menjadi persoalan menentukan lubang mana yang harus diambil dalam suatu keadaan pengambilan keputusan pada giliran pemain agar memaksimalkan jumlah biji congklak yang ada di dalam rumah.**

***Index Terms*—Congklak, Greedy, Program Dinamis, Strategi.**

I. PENDAHULUAN

Selain kegiatan kuliah, penulis juga mengikuti LSS (Lingkung Seni Sunda) sebagai kegiatan kemahasiswaan tambahan selain HMIF. Beberapa waktu yang lalu kepala bidang keangotaan LSS mengadakan home tournament di LSS. Salah satu cabang yang dilombakan adalah congklak. Dari sinilah penulis tertarik untuk mengimplementasikan apa yang sudah saya pelajari di mata kuliah Strategi Algoritma

II. PERMAINAN CONGKLAK

Congklak adalah salah satu jenis permainan tradisional. Congklak. Setiap daerah memiliki istilah sendiri untuk menyebtukan permainan ini. Di Jawa, permainan ini lebih dikenal dengan nama *congklak*, *dakon*, *dhakon* atau *dhakonan*. Di beberapa daerah di Sumatera yang berkebudayaan Melayu, permainan ini dikenal dengan nama *congkak*. Di Lampung permainan ini lebih dikenal dengan nama *dentuman lamban*, sedangkan di Sulawesi permainan ini lebih dikenal dengan beberapa nama: *Mokaotan*, *Maggaleceng*, *Aggalacang* dan *Nogarata*.

Permainan congklak dimainkan oleh 2 orang. Untuk memainkan congklak, dibutuhkan 2 komponen:

1. Biji congklak sejumlah 98 buah.
2. Papan dengan 16 buah lubang

Setiap pemain diberi sebuah lubang khusus pada papan yang merepresentasikan jumlah poin yang dia miliki. Lubang ini disebut rumah. Tujuan utama game ini adalah memiliki jumlah poin yang lebih banyak dibandingkan jumlah poin yang dimiliki lawan.

Pada kondisi awal, setiap lubang dalam papan diisi dengan 7 buah biji congklak kecuali lubang rumah milik masing masing pemain. Pada permulaanya, setiap pemain secara bersamaan memilih salah satu lubang di sisinya, kemudian mengambil sejumlah biji yang ada di dalam lubang itu hingga lubang itu kosong.

Secara bergantian pemain mengambil sejumlah biji congklak yang ada diantara 7 lubang di depanya kemudian membagi-bagikan biji yang ada ditangan berputar berlawanan arah jarum jam ke setiap lubang yang dilewati satu per satu kecuali lubang milik rumah lawan.

Ada berbagai varian aturan dalam permainan congklak sesuai dengan daerah congklak tersebut dimainkan. Berikut, sebagian aturan yang cukup familiar di lingkungan penulis (Sunda). Jika biji yang di tangan sudah habis maka ada 5 aturan:

1. Jika lokasi biji terakhir yang ditaruh ada di sisi lubang milik lawan dan lubang tersebut kosong, maka giliran pemain tersebut habis. Lawan mendapat giliran selanjutnya.
2. Jika lokasi biji terakhir yang ditaruh ada di sisi milik pemain yang mendapat giliran, lubang tersebut kosong dan di sisi lubang seberang yang bersesuaian (milik lawan) tidak terdapat biji congklak, maka giliran pemain tersebut habis. Lawan mendapat giliran selanjutnya.

3. Jika lokasi biji terakhir yang ditaruh ada di sisi milik pemain yang mendapat giliran, lubang tersebut kosong dan di sisi lubang seberang yang bersesuaian (milik lawan) terdapat biji congklak, maka ambil sejumlah biji congklak yang ada di lubang sisi lawan dan biji terakhir di lokasi tempat giliran pemain tersebut habis masukan ke dalam rumah milik pemain yang sedang mendapat giliran. Kondisi demikian disebut dengan "menembak". Kemudian giliran pemain tersebut habis, Lawan mendapat giliran selanjutnya.
4. Jika tempat lokasi lubang biji terakhir yang ditaruh terdapat biji, baik lokasi terakhir di sisi lawan atau di sisi pemain yang sedang mendapat giliran. Ambil sejumlah biji yang ada di dalam lubang tersebut (termasuk dengan biji terakhi yang diberikan) lanjutkan membagikan biji yang ada di tangan satu persatu berlawan arah jarum jam.
5. jika tempat lokasi lubang biji terakhir yang ditaruh adalah rumah sendiri, pemain berhak mendapat giliran lagi. Pemain bebas memilih salah satu diantara tujuh buah lubang yang tidak kosong yang ada di sisinya.[1]

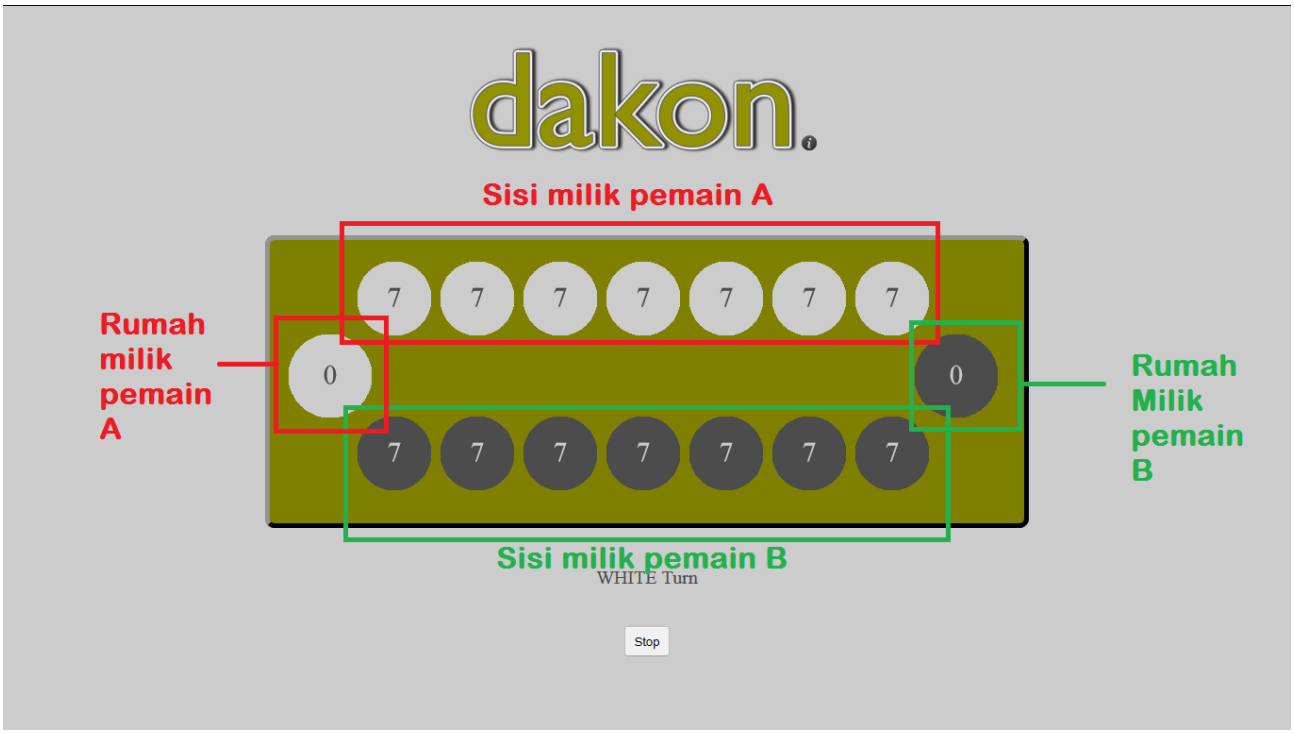
Permainan berakhir jika sudah tidak terdapat lagi biji diluar rumah. Kemudian skor pemain dihitung.

III. ALGORITMA PROGRAM DINAMIS

Program Dinamis adalah metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan langkah atau tahapan sedemikian sehingga solusi dari persoalan dapat dipandang dari serangkaian keputusan yang saling berkaitan. Pada penyelesaian persoalan dengan metode ini.

1. Terdapat sejumlah berhingga pilihan yang mungkin
2. Solusi pada setiap tahap dibangun dari hasil solusi tahap sebelumnya
3. Kita menggunakan persyaratan optimasi dan kendala untuk membatasi sejumlah pilihan yang harus dipertimbangkan pada suatu tahap.[2]

Tidak seperti algoritma greedy yang hanya mengambil solusi optimum pada tahap yang sedang aktif saja, algoritma program dinamis mempertimbangkan tahap yang sudah pernah diambil atau tahap yang akan diambil. Algoritma program dinamis juga tidak seperti exhaustive search yang mengenumerasi setiap pilihan, Algoritma program dinamis membuang himpunan solusi yang tidak mengarah ke solusi optimum.



Gambar 1 Diagram permainan congklak diambil dari salah satu website yang menyediakan permainan congklak, http://dakon-the-game.appspot.com/

Pada program dinamis, rangkaian keputusan yang

optimal dibuat dengan menggunakan prinsip optimalitas. Prinsip ini berbunyi jika solusi total optimal, maka bagian solusi sampai tahap ke-k juga optimal. Prinsip optimalitas berarti bahwa jika kita bekerja dari tahap ke k ke tahap ke-k+1, kita dapat menggunakan hasil optimal dari tahap k tanpa harus kembali ke tahap awal. Jika pada setiap tahap kita menghitung ongkos (cost), maka dapat dirumuskan bahwa

$$C(k+1) = C(k) + c(k,k+1)$$

Dimana,
C (k+1) : Cost pada tahap ke k+1
C(k) : Cost pada tahap ke k
c(k,k+1) : cost antara dari tahap k ke tahap k+1.

Persoalan program dinamis memiliki karakteristik tertentu yaitu :

1. Persoalan dapat dibagi menjadi beberapa tahap, yang pada setiap tahap hanya diambil satu keputusan.
2. Masing masing tahap terdiri dari sejumlah status (state) yang berhubungan dengan tahap tersebut. Secara umu, status merupakan bermacam kemungkinan masukan yang ada pada tahap tersebut. Jumlah status bisa berhingga atau tak berhingga.
3. Hasil keputusan yang diambil di setiap tahap ditransformasikan dari status yang berkaitan dengan satatus berikutnya pada tahap berikutnya.
4. Ongkos pada suatu tahap meningkat secara teratur dengan bertambahnya jumlah tahapan
5. Ongkos pada suatu tahap bergantung pada ongkos-ongkos tahap yang sudah berjalan.
6. Keputusan terbaik suatu tahap bersifat independen terhadap tahap keputusan yang dilakukan pad atahap sebelumnya.
7. Adanya hubungan rekursif yang mengidentifikasikan keputusan terbaik untuk setiap status pada tahap k memberikan keputusan terbaik untuk setiap status pada tahap k+1.
8. Prinsip optimalitas berlaku pada persoalan tersebut.

IV. ANALISIS PROGRAM DINAMIS DENGAN CONGKLAK

Persoalan congklak dapat dinyatakan dalam program dinamis dimana setiap tahapan direpresentasikan dengan persoalan bagaimana memilih lubang yang diambil untuk memaksimalkan jumlah akhir pada rumah. Akan didapatkan sebuah pohon keputusan kemungkinan yang diambil oleh pemain A / Pemain B secara bergantian dengan maksimal jumlah cabang anak 7 kemungkinan (7-tier tree).
Keadaan menjadi lebih kompleks karena beberapa hal :

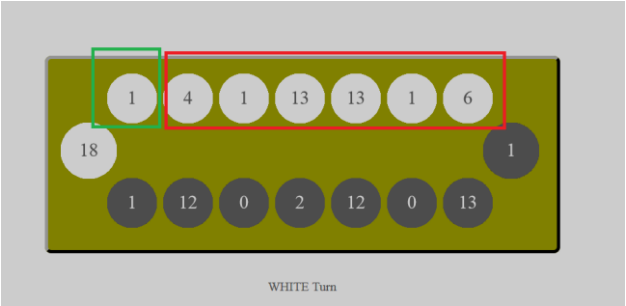
1. Suatu keadaan bergantung dari faktor yang tidak dapat diduga, yaitu keputusan yang diambil lawan dari giliran sebelum giliran pemain.
2. Tidak dapat dipastikan bahwa pohon keputusan yang dihasilkan akan berselang-seling. Misal di setiap dahan pada kedalaman ganjil, keadaan tersebut dapat dipastikan merepresentasikan keadaan yang diambil oleh pemain lawan. Karena sesuai aturan, seorang pemain mungkin mendapat beberapa giliran berturut-turut jika biji yang ia taruh terakhir ada di rumahnya.
3. Menjelang akhir permainan saat jumlah lubang kosong yang ada semakin banyak, kemungkinan pemain lawan / kita bermain lebih cepat dan lebih banyak kejadian "menembak" metode Greedy ***tidak mampu*** memaksimalkan kondisi tersebut. (Akan dibahas selanjutnya)

Aturan-aturan pergerakan biji dapat digambarkan dalam potongan pseudo code sebagai berikut:

Aturan dalam permainan Congklak

1. Pilih salah satu lubang yang ada di sisi sendiri.
2. Ambil seluruh biji yang ada di dalam lubang tersebut hingga tak tersisa.
3. Taruh sebuah biji dari tangan ke lubang sebelahnya berlawanan arah jarum jam, selain lubang rumah milik lawan. Jika biji di tangan belum habis, lakukan langkah 3 ini berulang.
4. Jika biji di tangan sudah habis cek apakah biji terakhir yang ditaruh ada di rumah milik pemain yang mendapat giliran

a. Jika ya, pemain tersebut mendapat giliran lagi, silahkan kembali ke langkah 1.
b. Jika tidak, cek apakah lokasi tempat menaruh biji terakhir kosong atau tidak
   i. Jika Tidak kosong ambil seluruh biji yang ada di dalam lubang tersebut, kembali ke langkah 3
   ii. Jika kosong, cek apakah lokasi terakhir berada di sisi lawan atau di sisi pemain yang mendapat giliran.
      1. Jika ada di sisi lawan, giliran pemain ini habis, lawan mendapat giliran
      2. Jika ada di sisi pemain yang aktif, cek apakah lubang di sisi seberang(milik lawan) yang bersesuian memiliki isi.
         a. Jika ya, ambil seluruh isi yang ada di lubang sisi seberang pindahkan ke rumah pemain yang mendapat giliran, lalu giliran pemain ini habis dan lawan mendapat giliran.
         b. Jika tidak, giliran pemain ini habis, lawan mendapat giliran



Gambar 2 Kondisi ketika algoritma greedy tidak lebih baik.

Perhatikan gambar 2, dalam game, putih (pemain A) memiliki giliran untuk bermain. Dapat dipastikan bahwa algoritma greedy akan merekomendasikan salah satu lubang yang berwarna merah bukan warna hijau. Karena algoritma greedy hanya mempertimbangkan jumlah biji terbesar yang bertambah ke rumah pada kondisi tersebut.(lubang yang ditandain dengan kotak warna hijau hanya memberikan satu buah pertambahan poin ke lubang rumah sementara lubang lainya yang ditandai warna merah, pasti memberikan lebih banyak pertambahan poin ke lubang rumah) . Padahal jika kita memilih lubang berwarna hijau terlebih dahulu, kita akan mendapat giliran tambahan untuk bermain sehingga pastilah jumlah poin yang didapatkan dalam kasus ini = jumlah poin yang dihasilkan dari algoritma greedy (hasil rekomendasi kotak yang ditandai warna merah) + 1. Jumlah ini lebih banyak dibandingkan jumlah yang dihasilkan algoritma greedy.

Oleh karena itu, algoritma yang pantas untuk menyelesaikan permasalahan penentuan lubang ke berapa yang harus diambil seoarang pemain untuk memaksimalkan jumlah poin yang didapat adalah algoritma program dinamis.

Untuk mentransformasi permasalahan ini menjadi suatu permasalahan yang dapat diselesaikan dengan algoritma program dinamis maka perlu diperhatikan aspek-aspek berikut.

- Karena terdapat unsur ketidakpastian dari faktor pemilihan lubang yang dilakukan oleh lawan, scope penggunaaan algorima program dinamis bukan seluruh game, tapi sebatas hingga giliran pemain habis (lokal). Sehingga output dari fungsi program dinamis adalah himpunan urutan nomor lubang yang harus diambil untuk menghasilkan kondisi maksimum pada kondisi ini saja. Bukan sebuah himpunan output lubang yang harus diambil dari awal permainan hingga akhir permainan (global).

- Solusi optimal yang diinginkan adalah bagaimana memaksimalkan jumlah petambahan biji pada lubang rumah.
- Untuk memodelkan kondisi papan congklak dibutuhkan dua buah larik A & B, sebuah fungsi yang mengembalikan nilai true jika biji terakhir jatuh di rumah pemain, sebuah fungsi yang menghasilkan himpunan nomor lubang yang bisa diambil pada giliran selanjutnya, sebuah fungsi untuk mendapatkan pertambahan poin pada rumah.
- Definisi fungsi rekursif nilai solusi optimal dapat dinyatakan sebagai berikut:

$$T(s)=\begin{cases} \max\{P(s)\} \text{ jika } \forall\, i \in s,\ F(i)=1 \\ \max\{P(s), P(s)+T(S'(s))\}, \text{ jika } \exists\, i \in s \mid F(s)\neq 1 \end{cases}$$

Dengan,
s : Himpunan nomor lubang yang bisa diambil pada tahap pengambilan keputsan saat itu.
P(s) : Fungsi untuk mencari jumlah poin yang bertambah untuk setiap s.
F(i) : Fungsi yang mengembalikan nilai true (1) jika apakah pengambilan lubang dengan nomor lubang-ke i berakhir di lubang rumah milik pemain, sehingga pemain mendapat giliran tambahan.
S'(s) : Fungsi yang megembalikan himpunan nomor lubang yang bisa diambil setelah pengambilan nomor lubang tertentu.

Berikut implementasi program untuk menentukan himpunan nomor lubang yang diambil pada suatu kondisi tertentu dalam pseudocode. Diasumsikan pada setiap permainan, kita mendapat giliran kedua untuk bergerak sehingga kita mampu melihat lubang mana yang dipilih oleh pemain lawan ketika mendapat giliranya.

```
Procedure Calculate(input
Playable_hole :array of integer [1
..N]output : solver : array of
integer)
{I.S : Playable_hole terdefinisi}
{F.S : Mengembalikan himpunan nomor
lubang yang harus diambil untuk
memaksimalkan jumlah skor di lubang
rumah pemain}

KAMUS LOKAL
Ditemukan : boolean
Max, hole: Array of integer
i : integer
{variabel Ditemukan : Digunakan
untuk menyatakan apakah terdapat
sebuah lubang dalam himpunan
playable hole sehingga biji
terakhir yang dibagikan dari tangan
ke papan ada di rumah pemain, max
untuk temporary variable. Hole
adalah temporary variable yang
berfungsi untuk mencari lubang
lubang yang dapat diambil lagi jika
pengambilansuatu lubang mampu
berakhir di rumah(mendapat giliran
tambahan lagi)}

{Diasumsikan seluruh fungsi yang
berkaitan dalam bagian algoritma
secara sepsifik sudah
diimplementasikan}

ALGORITMA
Ditemukan ← false
For(i=0; i<N; i++) do
  if(IsGetSecondChance(i)) do
      Ditemukan←true;

if(status) do
  for(i=0; i<N; i++)do
      FindMaximum(Profit(i))
      max.push_back(i);
      return max;
else
  for(i=0; i<N; i++)do
    if(IsGetSecondChance(i)) do
       hole←GeneratePlayableHole(i)
      FindMaximum(Profit(i))
      max.push_back(Compare(Profit(
i),Calculate(new_playable_hole)+Pro
fit(i)))
  return max;
```

IV. KESIMPULAN DAN SARAN

Dari uraian diatas, saya menyimpulkan beberapa hal.

1. Algoritma program dinamis dapat diterapkan untuk menyelesaikan persoalan penentuan lubang yang diambil untuk optimasi jumlah poin yang didapat dalam permainan congklak.
2. Algoritma Program dinamis mampu menyempurnakan algoritma greedy dalam

persoalan penentuan nomor lubang yang harus diambil untuk memaksimalkan jumlah poin dalam permainan congklak.

Hal ini dibuktikan dengan ditemukanya sebuah kasus dimana hasil optimasi menggunakan algoritma greedy tidak lebih baik dibandingkan dengan program dinamis.

REFERENSI

[1] http://www.wikihow.com/Play-Congkak diakses tanggal 19/12/13 pukul 17:00

[2] Munir, Rinaldi.2013. *Diktat Strategi Algoritma. Bandung. Halaman 190-192.*

[3] http://dakon-the-game.appspot.com/ diakses tanggal 19/12/13 pukul 20:00

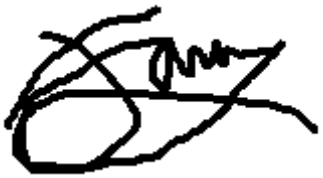
[4] *http://zerocrossraptor.wordpress.com/2012/04/26/count-and-capture-lets-play-congklak/ diakses tanggal 19/12/13 pukul 21:00*

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, 20 Desember 2012

ttd



Fawwaz Muhammad - 13511083