Penerapan Algoritma Runut Balik pada Permainan Kalah

Steven Andrew / 13509061
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13509061@std.stei.itb.ac.id

***Abstrak*—Makalah ini membahas bagaimana algoritma runut balik mencari dan menghasilkan urutan langkah-langkah yang mengakibatkan kondisi akhir tertentu pada permainan Kalah, yaitu pemain 1 akan menang atau lawannya yang menang. Meskipun pencarian membutuhkan kompleksitas waktu yang begitu besar, algoritma ini akan memberikan urutan langkah yang pasti untuk mencapai tujuan. Selain itu, algoritma runut balik dapat digunakan untuk menghitung rasio kemungkinan pemain 1 untuk menang, seri, dan kalah.**

***Index Terms* —Kalaha, runut balik.**

I. Pengenalan

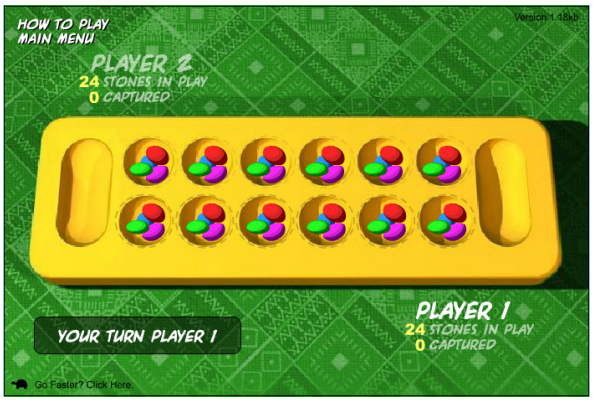
Kalah, biasa dikenal sebagai Mancala dan kadangkala disebut permainan "menabur" (*sowing*) atau "*count-and-capture*", adalah permainan papan yang ditemukan pada tahun 1940 dan diperkenalkan pada tahun 1950 oleh William Julius Champion Jr. [3] Permainan Kalah merupakan varian dari kelompok Mancala yang lebih dikenal. Varian permainan Mancala lainnya meliputi Congklak, Omweso, Ünee tugaluulakh, Bao, Sungka, Ayo, dan Igisoro.

II. Dasar-dasar Permainan Kalah

A. Spesifikasi Permainan Kalah

Kalah adalah permainan bergilir yang melibatkan dua pemain, yang dimainkan dengan menggunakan beberapa biji dan sebuah papan yang terdiri dari dua lubang besar yang disebut *kalahah* atau *store*, sebagai tempat tujuan untuk mengumpulkan biji, dan beberapa lubang yang disusun berjejer berpasangan, disebut *house* (biasanya 6 lubang per baris atau 2 × 6 keseluruhan).

Gambar 1. Papan permainan Kalah dari Afrika Barat.



Gambar 2. Keadaan awal permainan Kalah.

Pada awal permainan, biji-biji biasanya diletakkan masing-masing 4 pada setiap *house* dan setiap pemain menguasai keenam lubang tersebut pada satu baris. Tujuan dari permainan ini adalah mengumpulkan biji sebanyak-banyaknya pada *kalahah* milik pemain sendiri (sebelah ujung kanan pada arah pandangan pemain).

B. Aturan Permainan Kalah

Aturan-aturan pada berbagai varian permainan Mancala tidak begitu jauh berbeda namun prinsipnya sama. Berikut aturan-aturan pada permainan Kalah yang dibahas pada makalah ini.

1. Pada awal permainan, empat biji ditempatkan pada masing-masing 2 × 6 lubang (*house*), sehingga banyak biji keseluruhan yang digunakan adalah 48.
2. Setiap pemain mengendalikan 6 lubang di sisi papan dan *kalahah* pada samping kanan menurut arah pandangan.
3. Pemain mengambil langkah dengan menaburkan biji-biji (*sowing*). Pemain mengeluarkan semua biji dari satu lubang yang di bawah kendali, lalu menaruh tiap satu biji ke lubang-lubang lain berurutan secara berlawanan jarum jam, termasuk *kalahah* milik pemain sendiri, bukan milik lawan.
4. Jika biji yang terakhir ditabur bertempat pada *kalahah* pemain, pemain tersebut langsung mengambil langkah tambahan. Selain dari itu,

lawan akan mengambil langkah berikutnya.

5. Jika biji yang terakhir ditabur bertempat pada lubang kosong milik pemain, maka biji tersebut beserta seluruh biji pada lubang milik lawan di seberangnya akan dikumpulkan pada *kalahah* pemain.
6. Permainan berakhir jika seluruh lubang milik salah satu pemain kosong. Pemain yang menang memiliki jumlah biji pada kalahah dan semua lubang miliknya terbanyak.

III. Algoritma Runut Balik dan Penerapan pada Permainan Kalah

A. Algoritma Runut Balik

Bayangkan kita ingin memeriksa semua kamar pada sebuah bangunan besar dengan menelusurinya (mengunjunginya) satu per satu. Asumsikan kita dapat selalu mengingat kamar-kamar yang telah dikunjungi sehingga kita tidak akan mengunjunginya lagi. Lalu kita akan runut balik (*backtracking*) atau balik ke kamar yang tepat dikunjungi sebelumnya jika menemui jalan buntu (berarti pula kamar-kamar bertetangga telah dikunjungi).

Inilah bagaimana algoritma runut balik (*backtracking*) bekerja. Hasil penelusurannya membentuk pohon merentang (*spanning tree*). Algoritma ini merupakan bentuk lain dari algoritma DFS (*depth-first search*).

Algoritma runut balik dideskripsikan dalam langkah-langkah berikut. [2]

1. Set simpul awal sebagai simpul S.
2. Tandai simpul S telah dikunjungi.
3. Jika simpul S mempunyai tetangga yang belum dikunjungi, pilih salah satu tetangga yang belum dikunjungi dan set sebagai simpul S, lalu balik ke langkah 2; jika tidak, runut balik ke simpul yang dikunjungi sebelumnya dan balik ke langkah 2.

Kompleksitas waktu. Untuk graf yang ditelusuri tanpa perulangan, kompleksitas waktu algoritma runut balik adalah $O(|V| + |E|)$. Atau untuk graf dengan banyak tetangga simpul rata-rata w dan kedalaman penelusuran rata-rata d, kompleksitas waktunya adalah $O(w^d)$.

Kelebihan dan kekurangan dalam *pathfinding*. Algoritma runut balik membutuhkan memori lebih sedikit. Namun tidak menghasilkan jalur yang terpendek.

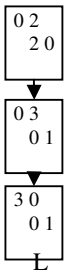
Algoritma runut balik diimplementasikan dalam bentuk iteratif dan rekursif. Algoritma dalam bentuk rekursif tidak membutuhkan data penyimpanan karena mekanisme rekursi sendiri telah melakukan "penyimpanan" parameter.

B. Pohon Penelusuran pada Permainan Kalah

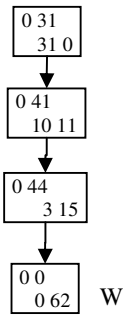
Dengan menggunakan algoritma runut balik, dapat dibentuk pohon penelusuran semua kemungkinan langkah-langkah dan status permainan.

Tidak sebatas pada permainan Kalah dengan 6 lubang dan 4 biji per lubang, secara umum anggap pada permainan terdapat m lubang pada tiap sisi dan n biji pada masing-masing lubang. Permainan dengan konfigurasi ini dinotasikan sebagai Kalah(m,n).

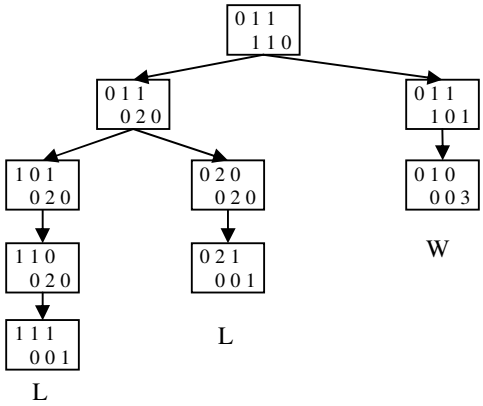
Di sini diberikan beberapa contoh pohon penelusuran pada Kalah(m,n) tertentu.



Gambar 3. Pohon penelusuran langkah-langkah pada Kalah(1,2).



Gambar 4. Pohon penelusuran langkah-langkah pada Kalah(2,1).



Gambar 5. Pohon penelusuran langkah-langkah pada Kalah(2,1).

C. Menghitung Distribusi Kemungkinan Menang, Kalah, dan Seri

Setiap daun pada pohon penelusuran (yang tidak mempunyai tetangga) merupakan keadaan akhir

permainan yang menentukan pemain manakah yang menang. Maka kita dapat menentukan kemungkinan pemain 1 menang, kalah, dan seri.

Pada contoh pohon penelusuran (Gambar 2 dan 3), dapat dihitung persentase kemungkinan menang/kalah/seri untuk Kalah(1,2) adalah 0/100/0, untuk Kalah(1,31) adalah 100/0/0, dan untuk Kalah(1,2) adalah 33/67/0, di mana pada kedua gambar tersebut, W menunjukkan pemain 1 menang, L menunjukkan pemain 1 kalah, dan D menunjukkan seri.

D. Mencari Alur Langkah untuk Memenangkan Permainan

Dengan menggunakan penunjuk simpul terkunjungi sebelumnya (status permainan), dapat dibentuk urutan langkah-langkah untuk setiap keadaan akhir permainan.

Hal ini dapat menentukan secara pasti cara untuk memenagkan permainan. Namun secara dinamis, jika diterapkan pada pemain yang menentukan langkahnya, tetap tidak dijamin untuk memenangkan karena langkah lawan dapat menyimpang dari dugaannya. Akan tetapi dapat dipilih langkah mana yang lebih berpeluang untuk menang. Dan hingga pada suatu keadaan, peluang menjadi pasti untuk menang atau kalah.

Jika ini diterapkan pada komputer, komputer akan "berpikir" lebih lama pada langkah-langkah awalnya, namun pada langkah berikutnya akan berpikir lebih cepat karena kompleksitas penelusurannya yang berkurang. Komputer dapat diprogram pula sehingga jika terdapat kepastian untuk kalah, komputer akan langsung menyerah.

Namun pada implementasi umumnya, agar komputer berpikir lebih cepat, algoritma runut balik tidak digunakan, melainkan taktik-taktik yang diterapkan dengan prinsip algoritma *greedy*.

IV. KESIMPULAN

Dalam mencari kemungkinan-kemungkinan secara pasti untuk memenangkan permainan Kalah, algoritma runut balik termasuk algoritma yang efektif dan lebih efisien, ketimbang cara yang lebih kasar yaitu *brute force*. Algoritma ini membutuhkan memori yang lebih sedikit pula.

Algoritma runut balik juga menghasilkan alur-alur langkah yang memenangkan salah satu pemain. Namun untuk mengimplementasikan komputer untuk memainkan permainan, algoritma ini secara praktis bukanlah pilihan.

DAFTAR PUSTAKA

[1] Irving, G., Donkers, H.H.L.M., and Uiterwijk, J.W.H.M., *Solving Kalah*. ICGA Journal, Vol. 23, No. 3, 2000, pp. 139-147.

[2] *http://en.wikipedia.org/wiki/Depth-first_search*, diakses pada tanggal 11/30/2010.

[3] *http://en.wikipedia.org/wiki/Kalah*, diakses pada tanggal 12/7/2011.

[4] *http://en.wikipedia.org/wiki/Mancala*, diakses pada tanggal 12/7/2011.

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, 9 Desember 2011

Steven Andrew / 13509061