ANALISA PENERAPAN ALGORITMA RUNUT - BALIK DALAM PENYELESAIAN PERMAINAN *SCRABBLE*

Twindania Namiesyva – NIM : 13505086

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if15086@students.if.itb.ac.id

Abstrak

Pada mulanya komputer diciptakan untuk mengerjakan beragam pekerjaan komputasi. Namun begitu, seiring dengan perkembangan teknologi, fungsi komputer juga mengalami perkembangan salah satunya adalah sebagai sarana hiburan. Diantara beragam hiburan yang ditawarakan oleh komputer, salah satu fasilitas hiburan yang akan selalu mendapat tempat di hati *user* adalah aplikasi permainan. Permainan menjadi sangat beragam jenisnya dan populer karena minat dari *user* sendiri. Permainan atau *game* tadinya hanya dimainkan untuk sekedar melepaskan penat dari pekerjaan. Tetapi sekarang menjadi sangat digemari karena beragamnya jenis fitur yang ditawarkan. User bisa memilih aplikasi permainan yang disukai berdasarkan kriteria yang diinginkan, misalnya petualangan, strategi, dan lain-lain. Pada makalah ini, penulis mencoba membahas tentang permainan *"Scrabble"* yang memungkinkan user untuk mencoba merangkai kata dari huruf – huruf yang tersedia dan menganalisa penerapan algoritma *runut - balik* sebagai salah satu cara penyelesaian persoalan pada ini. Melalui pembahasan dalam makalah ini, algoritma runut – balik digunakan pada proses perangkaian kata dari huruf – huruf yang tersedia. Jika dalam proses perangkaian kata ternyata terjadi ketidaksinambungan maka akan dilakukan proses *runut - balik* (runut balik).

***Kata kunci**: runut - balik, scrabble*

1. Pendahuluan

Permainan, baik yang tradisional maupun yang moderen (menggunakan teknologi atau komputerisasi) merupakan kegiatan yang disukai oleh hampir seluruh masyarakat di berbagai belahan dunia. Berbagai sebab dikemukakan sebagai alasan mengapa banyak orang menyukai permainan, diantaranya adalah: dapat membuat orang bersantai (*refreshing*), mengasah otak dan sebagina orang menyukai (khususnya komputer) karena desain dan fitur – fiturnya yang menarik. Salah satu bentuk yang sudah tidak asing lagi adalah komputer. disini merupakan aplikasi yang cukup diminati sebagai ajang *refreshing* bahkan olah otak. Bahkan tidak jarang, aplikasi pada komputer menimbulkan
kecanduan bagi *user* sehingga mengembangkannya sebagai hobi yang menantang.

Salah satu aplikasi permainan asah otak yang tersedia adalah "*Scrabble*". ini pada dasarnya merupakan suatu aplikasi yang memungkinkan user merangkai kata berdasarkan huruf- huruf yang tersedia.. Pada makalah ini, analisa penulis berkenaan dengan penyelesaian perangkaian kata yang dilakukan oleh *user*. Dalam proses ini, algoritma yang dipergunakan adalah algoritma runut - balik (runut balik) yang menangani ketidaksinambungan yang dapat terjadi jika kata yang dirangkai tidak sesuai.

1.1 Permainan Scrabble

Scrabble pada awalnya adalah permainan papan dan permainan menyusun kata yang dimainkan 2 atau 4 orang yang mengumpulkan poin berdasarkan nilai kata yang dibentuk dari keping huruf di atas papan permainan berkotak-kotak (15 kolom dan 15 baris).

Biji permainan berupa keping berbentuk bujur sangkar yang bertuliskan huruf pada salah satu sisi. Pemain mengambil hingga sebanyak tujuh buah keping huruf dari kantong, dan berusaha menyusun kata secara mendatar atau menurun seperti teka-teki silang. Kata-kata yang dibuat harus merupakan kata yang diizinkan untuk dimainkan berdasarkan kamus standar sesuai dengan bahasa yang dimainkan. Pemain yang mengumpulkan total poin tertinggi dinyatakan sebagai pemenang.

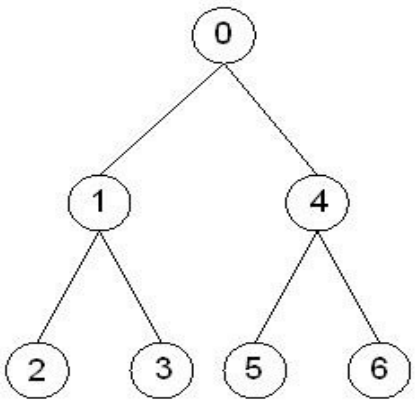
Permainan Scrabble mulanya diciptakan di tahun 1938 dengan nama "*Criss-Crosswords*" oleh seorang arsitek bernama Alfred Mosher Butts. Permainan ini merupakan penyempurnaan dari permainan *Lexiko* yang lebih dulu diciptakannya, tapi dilengkapi papan permainan dan cara bermain seperti teka-teki silang.

Pada makalah ini, penulis membatasi penyelesaian permainan scrabble hanya sampai pada perangkaian kata untuk setiap tujuh huruf yang telah ditentukan sebelumnya sebagai huruf – huruf yang harus disusun.

1.2 **Algoritma Runut - Balik**

Algoritma runut - balik merupakan perbaikan algoritma brute force dengan berbasis *DFS* untuk mencari solusi persoalan di antara semua kemungkinan solusi yang ada. Algoritma ini hanya mempertimbangkan pencarian yang mengarah ke solusi, sehingga seringkali menghemat waktu pencarian. Algoritma ini banyak diterapkan untuk program *games* dan permasalahan pada bidang kecerdasan buatan.

Seperti yang telah dijelaskan diatas bahwa pencarian solusi dengan menggunakan algoritma runut - balik ini berbasis pada *DFS*, maka kita menggunakan pohon ruang status.



Gambar 1 Contoh Pohon Ruang Status

Langkah – langkah pencarian solusi:

1. Membentuk lintasan dari akar ke daun dengan metode DFS. Simpul dinomori sesuai dengan urutan kelahirannya.
2. Untuk setiap lintasan yang tidak mengarah ke solusi, matikan simpulnya.
3. Jika pembentukan lintasan berakhir dengan simpul mati, maka proses pencarian dilakukan dengan membangkitkan simpul anak yang lainnya. Bila seluruh simpul anak sudah tidak dapat dibangkitkan, maka pencarian solusi dilanjutkan dengan melakukan runut - balik ke simpul hidup terdekat.
4. Pencarian dihentikan jika telah ditemukan solusi atau sudah tidak ada lagi simpul hidup untuk runut - balik

2. Skema Umum Algoritma Runut – Balik

Berikut ini adalah algoritma runut - balik[2] dengan menggunakan algoritma rekursif.

```
procedure RunutBalikR(input k:integer)
{Mencari semua solusi persoalan dengan metode
runut-balik; skema rekursif
Masukan: k, yaitu indeks komponen vektor solusi,
x[k]
Keluaran: solusi x = (x[1], x[2], ..., x[n]) }

Algoritma:
 for tiap x[k] yang belum dicoba sedemikian
 sehingga
      ( x[k]←T(k)) and B(x[1], x[2], ... ,x[k])=true do
   if (x[1], x[2], ... ,x[k]) adalah lintasan dari akar
   ke daun then
        CetakSolusi(x)
   endif
   RunutBalikR(k+1) {tentukan nilai untuk x[k+1]}
 endfor
```

$T(k)$ membangkitkan nilai untuk x_k, yang merupakan komponen vektor solusi. $B(x_1, x_2, \ldots, x_k)$ adalah fungsi pembatas menentukan apakah $(x_1, x_2, \ldots, x_k)$ mengarah ke solusi .

3. Alasan Penggunaan Algoritma Runut – Balik

Penulis menganalisa pencarian solusi untuk permainan ini dengan algoritma runut - balik karena beberapa alasan, antara lain :
1. Tidak memiliki informasi yang cukup untuk mengetahui huruf pertama mana yang harus apa yang harus dipilih.
2. Setiap keputusan yang diambil mengarah pada sekumpulan pilihan yang baru.
3. Beberapa pilihan yang ada kemungkinan merupakan solusi dari permainan ini.

Dengan adanya pilihan yang banyak ini membuat kebanyakan pemain memilih untuk melakukan pilihan secara *brute force,* yang artinya memilih acak atau terserah dari si pemain. Namun jarang atau sedikit sekali pilihan tersebut menuju pada solusi, karena keterbatasan waktu yang dimiliki oleh si pemain.

3. Penyelesaian Permainan Scrabble dengan Algoritma Runut – Balik

Untuk menyelesaikan permainan scrabble dengan menggunakan algoritma runut – balik, ada beberapa langkah yang harus dilakukan.

1. Daftar semua huruf yang akan dirangkai sebagai kata.
2. Pilih salah satu huruf sebagai huruf awal (huruf ke-k).

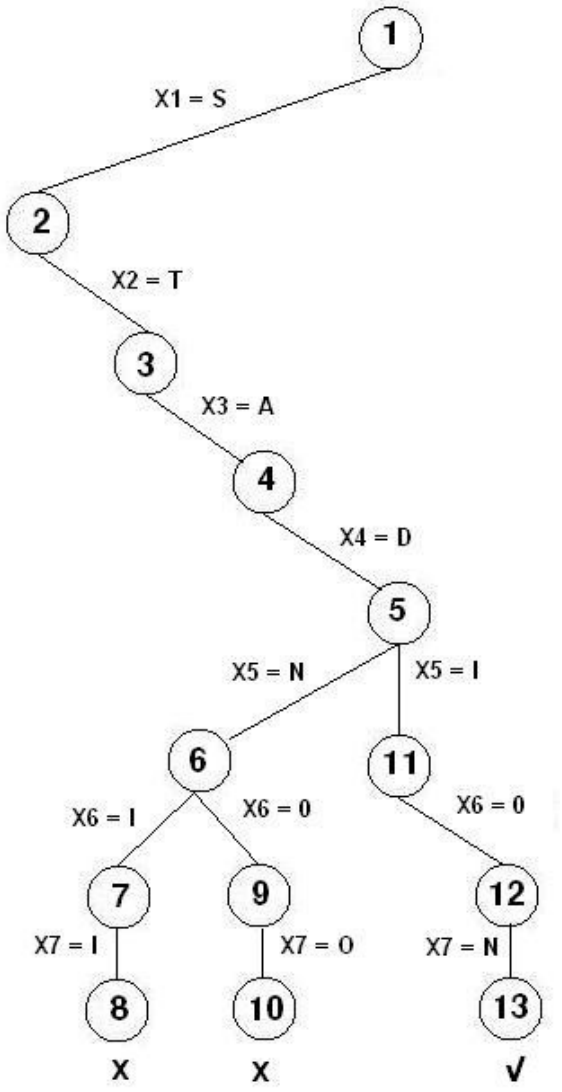
3. Pilih huruf berikutnya untuk kemudian dirangkai berurutan dengan huruf sebelumnya sebagai huruf ke-k+1.
4. Cek apakah rangkaian huruf sementara mengarah ke solusi (pada kasus ini maksudnya apakah rangkaian huruf sementara merupakan subset dari kata yang diizinkan untuk dimainkan berdasarkan kamus standar sesuai dengan bahasa yang dimainkan).
5. Cek apakah rangkaian huruf sementara merupakan solusi (pada kasus ini maksudnya apakah rangkaian kata sementara yang dibentuk merupakan salah satu diantara kata - kata yang diizinkan untuk dimainkan berdasarkan kamus standar sesuai dengan bahasa yang dimainkan)
6. Jika rangkaian huruf sementara mengarah ke solusi, maka lanjutkan langkah 3, dan 4.
7. Jika rangkaian huruf sementara tidak mengarah ke solusi, maka bangkitkan huruf ke-k+1 yang baru. Bila sudah tidak ada huruf tersisa untuk dibangkitkan, maka bangkitkan huruf ke-k yang lain.
8. Jika sudah tidak ada huruf tersisa untuk dibangkitkan atau sudah ditemukan rangkaian kata yang merupakan solusi, maka pencarian solusi dihentikan.

Adapun algoritmanya secara garis besar untuk memecahkan permasalahan ini adalah seperti yang ada dibawah ini.

```
if bukan solusi
   then
      while mengarah pada solusi do
         if terdapat huruf tersisa sedemikian
            sehingga huruf tersebut dapat dirangkai
            dengan kata sebelumnya
         then
            tambahkan satu huruf pada rangkaian kata
         if tidak terdapat huruf tersisa untuk
            dibangkitkan dan rangkaian kata sementara
            bukanlah solusi
         then
           backtrack
         endif
         endif
      endwhile
else
      solusi {semua kotak telah diinjak dan posisi
      kesatria kembali ke kotak awal}
```

Seperti telah disebutkan sebelumnya pada subbab 1.1, penulis membatasi penyelesaian permainan *scrabble* ini hanya sampai pada perangkaian kata untuk setiap tujuh huruf yang telah ditentukan sebelumnya sebagai huruf – huruf yang harus disusun. Pembatasan itu tidak hanya berarti jumlah huruf yang tersedia untuk disusun hanya tujuh, namun juga solusi kata yang dicari mempunyai ukuran tujuh huruf. Pemilihan jumlah tujkuh buah huruf sebagai jumlah huruf yang tersedia berdasarkan aturan permainan, sedangkan penetapan aturan untuk menghasilkan solusi dengan ukuran tepat tujuh buah huruf adalah batasan khusus dari penulis sendiri untuk memudahkan penerapan algoritma runut - balik. Pada permainan *scrabble* yang asli, ukuran kata solusi tidak diharuskan sejumlah tujuh buah.,

Di bawah ini adalah gambar pohon ruang status untuk permainan *scrabble* contoh1, dimana daftar huruf untuk dirangkai pada contoh1 adalah S, T, A, D, N, I, O.



Gambar 2 Pohon Ruang Status Contoh1

5. Kesimpulan

Penggunaan algoritma *bactracking* dalam mencari solusi untuk permainan ini memperbaiki pencarian solusi dengan algoritma *brute force.* Dalam

pencarian solusi, kita menggunakan pohon ruang status dengan basis *DFS (Depth First Search),* terus menelusuri simpul akar sampai simpul anak, sampai menemukan solusi atau tidak ada lagi simpul hidup yang dapat ditelusuri. Dengan pohon ruang status, kita berhenti saat kita menemukan solusi dari semua kemungkinan solusi yang ada. Sehingga metode ini jauh lebih efektif dibandingkan dengan *brute force* yangmencari semua kemungkinan solusi.

Saran yang dapat diberikan adalah coba untuk menemukan solusi dengan menggunakan algoritma lainnya (*greedy, BFS, divide and conquer*), mungkinakan lebih efektif dibandingkan dengan algoritma runut - balik ini.

DAFTAR PUSTAKA

[1] B, Dian. (2002). Scrabble, wujud rasa cinta Butts.

http://www.korantempo.com/news/2002/2/17/Idea/30.html. Tanggal akses: 22 Mei 2007 pukul 11:15.

[2] Munir, Rinaldi. (2004). Bahan Kuliah IF5054 Kriptografi. Departemen Teknik Informatika, Institut Teknologi Bandung.

[3] *http://id.wikipedia.org/wiki/Scrabble*. Tanggal akses: 22 Mei 2007 pukul 11:16.