APLIKASI ALGORITMA PENCOCOKAN STRING DALAM PERMAINAN SCRABBLE

Rahmat Izwan Heroza

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
e-mail : if15008@students.if.itb.ac.id

ABSTRAK

Scrabble merupakan permainan yang populer. Pada permainan ini, kita akan mempunyai sejumlah huruf yang acak. Dengan huruf-huruf itu, kita harus menyusun suatu kata. Tiap-tiap huruf mempunyai nilai yang berbeda-beda. Nilai dari tiap-tiap huruf dijumlahkan, yang nantinya akan dianggap sebagai point jalan pemain yang bersangkutan. Pada papan juga terdapat faktor-faktor yang dapat meningkatkan jumlah point yang bisa dicapai. Sebagai contoh, terdapat ”double letter” pada sel papan yang berarti mengalikan nilai yang terdapat pada huruf yang ada di ata sel tersebut dengan 2. selain itu juga terdapat ”triple word” pada sel papan yang berarti mengalikan jumlah nilai dari tiap-tiap huruf dengan 3.

Jika scrabble dibuat menjadi program, maka permasalahan utamanya adalah bagaimana kita mengetahui bahwa kata yang diajukan oleh pemain benar atau salah. Kita akan mempunyai suatu pustaka acuan, sebut saja kamus scrabble, yang akan menjadi sumber pembanding. Jika kita berbicara masalah membandingkan dua buah string, maka kita akan berbicara tentang algoritma pencocokan string. Algoritma yang memeriksa apakah satu string merupakan subset dari string yang lain, ataukah kedua string itu persis sama. Dalam permainan scrabble, suatu string dikatakan memenuhi syarat hanya jika string yang diajukan, terdapat dalam kamus scrabble. Ini berarti kita akan membandingkan dengan cara yang kedua, yaitu membandingkan apakah string yang diajukan persis sama dengan salah satu string yang terdapat di dalam kamus scrabble.

Bagaimanakah algoritma perbandingan/pencocokan string di dalam aplikasinya pada permainan scrabble? Suatu hal yang perlu dijelaskan. Oleh karena itulah, kami merasa perlu untuk mengangkat masalah algoritma pencocokan string pada permainan scrabble ini dalam makalah kami

***Kata kunci**: brute force, subprogram, body, overloading, prosedur, fungsi,*

1. PENDAHULUAN

Scrable adalah permainan menyusun kata. Jika permainan ini akan dibuat menjadi suatu program, maka setidaknya akan terdapat 3 sistem yang akan terlibat. Sistem yang pertama adalah user. Dalam hal ini, user adalah manusia yang memainkan permainan scrable ini. User akan akan menyusun sebuah kata dari huruf-huruf yang dimilikinya. Sistem yang kedua adalah komputer. Maksudnya disini adalah pemain kedua yang berperan sebagai musuh daripada user. Komputer akan menyusun sebuah kata dari huruf-huruf yang dimilikinya dengan menggunakan suatu algoritma tertentu. Awalnya user akan mencoba menyusun segala kemungkinan kata dari huruf-huruf yang dimiliknya dengan algoritma Brute Force sambil mencocokkan kata yang disusunnya itu dengan kata-kata yang terdapat dalam pustaka permainan menggunakan algoritma pencocokan string.

Sistem yang ketiga adalah juri. Juri bertugas memeriksa apakah kata yang disusun oleh user cocok dengan salah satu kata yang terdapat dalam kmaus scrabble atau tidak dengan menggunakan algoritma Pencocokan String. Adapun kata yang disusun oleh komputer tidak akan diperiksa oleh juri karena kedua sistem ini menggunakan algoritma yang sama.

Algoritma Pencocokan String memiliki banyak metode. Masing-masing metode memiliki keunggulan tersendiri. Dan juga masing-masing metoda itu harus digunakan dalam masalah yang berbeda pula. Oleh karena itu kita harus berhati-hati dalam memilih metode yang akan kita gunakan. Dalam persoalan pencocokan string dalam permainan scrabble, algoritma pencocokan string yang tepat untuk digunakan adalah algoritma brute force dengan sedikit penyesuaian.

Agar program menjadi mudah untuk dipelajari, maka program dibuat dalam bentuk modular. Prosedur-prosedur atau fungsi-fungsi yang memiliki tujuan khusus akan dibuat dalam bentuk sub program.

Gambar papan scrabble

2. KAMUS SCRABBLE

Kamus scrabble adalah kamus yang digunakan sebagai pustaka acuan dalam permainan scrabble. Sah tidaknya suatu kata yang diajukan oleh pemain ditentukan oleh kamus kata ini. Jika kata yang diajukan terdapat dalam kamus kata, maka kata yang diajukan boleh dipasang di atas papan permaianan scrabble (sah). Jika tidak, maka kata yang diajukan ditolak dan tidak boleh dipasang di atas papan permainan scrabble (tidak sah). Untuk menyederhanakan persoalan dan agar algoritma pencocokan string yang kita buat lebih mangkus, kita asumsikan bahwa kata-kata yang terdapat dalam kamus scrabble ini, tersusun secara alphabetis.

Pada kamus scrabble, terdapat kursor pemindai karakter, sebut saja CC yang berisi karakter pada posisi kursor. Terdapat prosedur **nextChar** yang berfungsi memajukan CC satu karakter. Selain itu juga terdapat prosedur **nextKata.**

3. ALGORITMA PENCOCOKAN STRING

Algoritma Pencocokan String adalah suatu algoritma memeriksa kesamaan pettern suatu kata dengan kata yang terdapat dalam kamus kata. Dua buah kata dikatakan cocok apabila jumlah huruf kedua kata itu sama dan huruf-huruf pada indeks yang bersesuaian bernilai sama. Terdapat 3 macam algoritma pencocokan string yang lazim digunakan yaitu algoritma brute force, algoritma Knuth-Morris-Pratt (KMP), dan algoritma Boyer-Moore (BM).

Algoritma yang pertama adalah Algoritma Brute Force Pencocokan string. Kita asumsikan bahwa kamus data (teks) berada di dalam array T[1..n] dan *pattern* yang akan dicocokkan berada di dalam array P[1..m], maka algoritma brute force pencocokan string adalah sebagai berikut:

1. Mula-mula pettern P dicocokkan pada awal teks T.
2. jika cocok, dengan bergerak dari kiri ke kanan, bandingkan setiap karakter di dalam pattern P dengan karakter yang bersesuaian di teks T sampai:
    a. semua karakter yang dibandingkan cocok, atau
    b. ditemukan ketidakcocokan (belum berhasil).
3. Bila pattern P belum ditemukan kecocokannya dan teks T belum habis, geser pattern P satu karakter ke kanan dan ulangi langkah 2.

Algoritma yang lain adalah KMP. Algoritma ini memiliki kompleksitas waktu yang lebih baik dari pada algoritma Brute Force. Pada algoritma brute force, setiap kali ditemukan ketidakcocokan pattern dengan teks, maka pattern digeser satu karakter ke kanan. Sedangkan pada algoritma KMP, kita memelihara informasi yang digunakan untuk melakukan pergeseran yang lebih jauh. Algoritma ini mangkus, hanya jika persoalan pencocokan string yang kita hadapi berupa mencari substring suatu string terhadap string yang lain

Lain lagi dengan algoritma Boyer-Moore yang merupakan variasi lain dari pencarian string dengan melompat maju sejauh mungkin. Algoritma BM melakukan perbandingan pattern mulai dari kanan (belakang), dimana algoritma KMP melakukan perbandingan mulai dari kiri (depan). Sama hal nya seperti algoritma KMP, algoritma ini mangkus, hanya jika persoalan pencocokan string yang kita hadapi berupa mencari substring suatu string terhadap string yang lain.

Seperti yang dijelaskan pada bagian pendahuluan, kita harus memilih dengan tepat, metode pencocokan string mana yang akan kita pakai dalam menyelesaikan masalah ini. Algortima KMP dan BM memang lebih mangkus jika dibandingkan dengan algoritma brute force. Tapi itu berlaku hanya jika persoalan pencocokan string yang kita hadapi, berupa mencari substring suatu string terhadap string yang lain. Sedangkan masalah pencocokan string dalam permainan scrabble tidak demikian. Suatu kata dikatakan sah, apabila kata tersebut persis sama dengan salah satu kata yang terdapat di dalam kamus scrabble, dengan artian jumlah kata dan huruf-huruf yang bersesuaian pada kedua string persisi sama.

Oleh karena itu algoritma pencocokan string yang akan dipakai dalam permainan scrabble ini adalah Algoritma Brute Force Pencocokan string dengan sedikit penyesuaian.

Kita asumsikan bahwa kamus scrabble (teks) berada di dalam array T[1..n] dan *pattern* yang akan dicocokkan

berada di dalam array P[1..m], maka algoritma brute force pencocokan string yang telah disesuaikan dengan kasus permainan scrabble adalah sebagai berikut:

1. Mula-mula pettern P dicocokkan pada awal teks T.
2. jika cocok, dengan bergerak dari kiri ke kanan, bandingkan setiap karakter di dalam pattern P dengan karakter yang bersesuaian di teks T sampai:
    a. semua karakter yang dibandingkan cocok, atau
    b. ditemukan ketidakcocokan (belum berhasil).
3. Bila pattern p belum ditemukan kecocokannya dan teks T belum habis, geser pattern P menuju awal kata selanjutnya di dalam teks (ditandai dengan "spasi") dan ulangi langkah 2. Bila semua karakter yang dibandingkan cocok, periksa apakah teks sudah mencapai akhir kata atau belum. Jika sudah, berarti pencarian sukses, jika belum berarti gagal, geser pattern P menuju awal kata selanjutnya di dalam teks, ulangi langkah 2.

4. ALGORITMA BRUTE FORCE

Brute Force adalah sebuah pendekatan yang lempang (straightforward) untuk memecahkan suatu masalah, biasanya didasarkan pada pernyataan masalah (problem statement) dan definisi konsep yang dilibatkan. Algoritma brute force memecahkan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas (obvious way).

Dalam persoalan permainan scrabble, algoritma ini adalah pilihan yang tepat. Karena ketidakpastian pola pada kamus kata, maka mau tidak mau, kita harus mencoba segala kemungkinan susunan kata, brute force.

5. ALGORITMA SCRABBLE

Seperti yang telas dijelaskan diatas, algoritma yang digunakan dalam permainan scrabble ini merupakan gabungan dari algoritma brute force dan algoritma pencocokan string.

Karena algoritma yang akan dipakai oleh juri hanyalah algoritma pencocokan string sedangkan algoritma yang akan dipakai oleh komputer adalah algoritma brute force dan pencocokan string, maka agar lebih efisien, kedua algoritma ini (pencocokan string dan brute force) akan dibuat dalam sub program yang terpisah.

5.1. Prosedur nextKata

Kita akan menerapkan *overloading* dalam prosedur ini. *Overloading* berarti, kita mendefinisikan lebih dari satu *body* terhadap *subprogram* yang sama. Yang pertama, prosedur nextKata tanpa parameter, (**nextKata()**) akan meletakkan kursor scanning terhadap kamus scrabble pada kata selanjutnya. Hal ini bisa diketahui dengan mencari karakter spasi dan meletakkan kursor pada huruf setelahnya. Kemudian definisi yang kedua, prosedur nextKata dengan parameter, (**nextKata(char karakterAwal)**) akan meletakkan kursor scanning terhadap kamus scrabble padakata pertama yang berawalan **karakterAwal**.

Ada beberapa langkah dalam prosedur **nextKata()**, yaitu:

1. Scanning kamus scrabble sampai menemukan karakter *space*.
2. Majukan kursor satu huruf untuk menemukan hurf pertama dari kata berikutnya.

Begitu pula dengan algoritma **nextKata(char karakterAwal).** Langkah-langkah nya adalah:

1. Scanning kamus scrabble sampai menemukan karakter *space*.
2. Majukan kursor satu huruf untuk menemukan hurf pertama dari kata berikutnya.
3. Jika huruf yang sedangkan di scan adalah **karakterAwal,** maka berhenti. Jika tidak, ulangi dari langkah 1.

5.2. Fungsi IsCocok

Inilah inti dari permasalahan permainan scrabble. Di dalam fungsi **IsCocok**, kita akan mengimplementasikan algoritma pencocokan string. Langkah-langkah nya adalah seperti yang telah dijelaskan pada bab 3.

Fungsi **IsCocok** adalah fungsi yang mengembalikan boolean. Akan mengembalikan *true* jika kedua string yang dicocokkan sama. Dan akan mengembalikan *false* jika kedua string yang dicocokkan tidak sama.
Kita asumsikan bahwa kamus scrabble (teks) berada di dalam array T[1..n] dan *pattern* yang akan dicocokkan berada di dalam array P[1..m]. Untuk pertama kali posisi kursor di kamus scrabble berada pada huruf pertama kata pertama. Maka langkah-langkah dalam fungsi **IsCocok** adalah sebagai berikut:

1. Panggil prosedur **nextKata(P[1]),** yang berarti kita akan menuju huruf pertama dari kata pertama yang berawalan P[1] dalam kamus scrabble. Indeks =1;
2. Cocokkan CC dengan P[Indeks]
3. Jika cocok, majukan CC dan incrementasi Indeks. Ulangi langkah 2 sampai karakter pada pattren habis.
4. Jika cocok dan karakter pada pattern habis, panggil prosedur **nextChar**() , periksa apakah CC sama dengan *space*. Jika ya, berarti cocok, kembalikan true. Proses berhenti.

5. Jika tidak cocok, panggil fungsi **nextKata().** Jika CC sama dengan **karakterAwal**, ulangi dari langkah 2. Jika CC tidak sama dengan **karakterAwal,** proses berhenti, kembalikan false

```
function IsCocok(P array [1..n] of char) -> boolean
{
    mengembalikan true jika pattern yang terdapat
    dalam array T cocok dengan salah satu kata yang
    terdapat dalam kamus scrabble
}
Deklarasi:
    CC : char;
    habis : boolean;
    Indeks : integer;
    procedure nextChar();
    procedure nextKata();
    procedure nextKata(char c);

Algoritma:
    nextKata(P[1]);
    do
      while((CC==P[Indeks]) and (!habis)) then
            nextChar();
            Indeks++;
      endwhile
      if (habis) then
            nextChar();
            if (CC==' ') then
                  return true;
                  stop
            endif
      nextKata();
    while (CC==P[1]);
    return false;
```

6. KESIMPULAN

Permainan scrabble menggunakan algoritma pencocokan string sebagai jiwa dari programnya. Algoritma pencocokan string yang tepat digunakan dalam permainan scrabble adalah algoritma brute force pencocokan string.

REFERENSI

[1] *Munir, Rinaldi*. 2005. "Strategi Algoritmik". Teknik Informatika: Bandung.

[2] *www.scrabble.com.* Waktu akses: 22 mei 2007, 22:00 WIB.