Algoritma Boyer-Moore dalam Pencarian String

Imam Sulistyo[1], Adie Pradipto[2], Adi Setia Perwira[3]

Departemen Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail: smami@students.math.itb.ac.id[1], adie@students.itb.ac.id[2],
adisp@students.itb.ac.id[3]

Abstrak

Pencarian string merupakan masalah yang lazim ditemui, terutama penacrian string pada teks. Sudah banyak algoritma diciptakan untuk menyelesaikan permasalahan ini, salah satunya adalah algoritma Boyer-Moore. Algoritma yang diciptakan oleh R.M Boyer dan J.S Moore ini terkenal karena banyak diterapkan pada algoritma pencocokan untuk banyak string (*multiple pattern*). Makalah ini akan membahas mengenai algoritma Boyer-Moore secara singkat dan gamblang disertai dengan contoh.

Kata kunci: Boyer-Moore, pencarian string.

1. Pendahuluan

Setiap hari manusia selalu berurusan dengan data dalam bentuk yang berbagai macam, namun data berupa teks merupakan data yang paling umum digunakan untuk pertukaran informasi dan manfaat lainnya. Seringkali operasi pada teks melibatkan proses pencarian terhadap kemunculan suatu string dan lokasinya pada teks yang bersangkutan. Proses seperti ini sangatlah penting dalam mengatasi berbagai masalah, termasuk *text editing*.

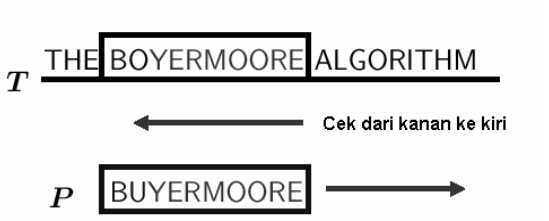
Pencarian string terfokus untuk menemukan kemunculan suatu string yang terdiri dari **m** karakter(selanjutnya disebut *pattern*) dalam suatu teks sepanjang **n** karakter. Dalam proses ini diperlukan algoritma sehingga letak *pattern* tersebut pada teks bisa diketahui. Banyak sekali algoritma yang telah diciptakan untuk menyelesaikan permasalahan ini. Setiap algoritma berusaha untuk menghindari masalah yang muncul pada algoritma yang telah ada dan menjadi yang terbaik. Salah satunya adalah algoritma Boyer-Moore.

2. Konsep Dasar Boyer-Moore

Sebelum algoritma Boyer-Moore, algoritma brute-force yang ada masih sederhana. Algoritma paling sederhana dan mungkin juga algoritma pertama yang ditemukan ini prinsipnya adalah mencocokan setiap karakter dalam *pattern* dengan karakter pada teks dari kiri ke kanan. Jika menemui ketidakcocokan, maka pattern dipindahkan ke kanan sejauh satu karakter dan mengulang proses pencocokan sampai *pattern* ditemukan atau mencapai akhir dari teks.

Algoritma Boyer-Moore diciptakan oleh R.M Boyer dan J.S Moore. Algoritma ini terkenal karena banyak diterapkan pada algoritma pencocokan untuk banyak string (*multiple pattern*).

Ide dasar dari algoritma Boyer-Moore (BM) sebetulnya sama sederhananya dengan algoritma brute-force. Algoritma ini merupakan improvisasi dari KMP (Knuth-Moris-Pratt), algoritma yang menghilangkan proses pengecekan ulang oleh algoritma brute-force dengan menggunakan bantuan tabel berisi langkah perpindahan pattern yang harus dilakukan ke kanan ketika menjumpai ketidakcocokan karakter. Pada algoritma BM, pencocokan dilakukan dengan mengubah arah pencocokan, yaitu mencocokan karakter dimulai dari kanan (karakter terakhir pada *pattern*) ke kiri. Sementara itu pattern tetap digeser ke kanan.



Jika menemui ketidakcocokan, pattern akan dipindahkan ke kanan dan jika mungkin, membuat kecocokan dengan karakter pada teks yang sedang dilakukan pengecekan.

3. Penerapan Algoritma Boyer-Moore

Dalam penerapannya algoritma Boyer_Moore membutuhkan 2 syarat awal, yaitu teks dan *pattern*. Misalkan terdapat teks yang berisi : ‘mau nyari yang dicari’ dan suatu *pattern* ‘cari’:

```
mau nyari yang dicari
   |
cari
```

Karakter pertama yang diperiksa adalah antara ‘i’ dengan karakter spasi pada teks. Karena tidak cocok dan tidak ada karakter spasi pada pattern, maka pattern bisa dipindahkan sejauh panjang pattern (dalam contoh ini 4 karakter).

```
mau nyari yang dicari
       |
    cari
```

Ketidakcocokan muncul kembali. Namun, kali ini karakter r terdapat pada pattern. Pindahkan pattern sehingga karakter ‘r’ yang terdapat paling kanan pada pattern cocok dengan karakter ‘r’ pada teks.

```
mau nyari yang dicari
        |
     cari
```

Karena ‘y’ tidak sama dengan ‘c’ maka pindahkan ke kanan sebanyak 4 karakter (karena tidak ada lagi karakter ‘i’ pada pattern selain ujung kanan).

```
mau nyari yang dicari
           |
         cari
```

Tidak ada ‘n’ pada pattern, pindah 4 karakter.

```
mau nyari yang dicari
              |
            cari
```

Karakter ‘i’ sama, tapi karakter ‘d’ dan ‘r’ tidak. Pindah lagi 4 karakter

```
mau nyari yang dicari
                  |
                cari
```

Pattern ditemukan.

Dari contoh di atas, diperlukan 12 kali pencocokan. Pattern ditemukan pada karakter ke-18 pada teks.

Pada algoritma ini, jika karakter pada teks tidak ada di dalam pattern, maka pattern bisa dipindahkan sejauh **m** karakter (panjang karakter). Jika karakter yang sedang diperiksa (misal ‘x’) pada teks terdapat di dalam pattern, maka pindahkan pattern sehingga karakter yang paling kanan (‘x’ yang letaknya paling kanan pada pattern) cocok dengan karakter pada teks.

4. Ringkasan Algoritma Boyer-Moore

Secara umum, kita dapat meringkas prosedur yang telah dijelaskan menjadi sebuah algoritma Boyer Moore sebagai berikut:

Pertama, dilakukan proses awal pemeriksaan karakter terhadap pattern dan teks. Kemudian dibuat indeks dari kemunculan karakter-karakter dalam pattern dan teks tadi. Indeks ini dimasukkan ke dalam suatu tabel tambahan.

Tabel dapat diimplementasikan sebagai *array* dua dimensi. Pada baris, diisikan karakter yang muncul pada *pattern* sedangkan pada kolom diisikan dengan karakter yang muncul pada teks.

Berdasarkan contoh di atas, bisa dibuat tabel sebagai berikut

	c	a	r	i
m	4	4	4	4
a	4	0	1	2
u	4	4	4	4
’spasi’	4	4	4	4
n	4	4	4	4
y	4	4	4	4
r	4	4	0	1
d	4	4	4	4
i	4	4	4	0
c	0	1	2	3

Angka-angka pada tabel merupakan langkah yang harus dilewati oleh *pattern* (dalam satuan karakter). Misalkan dalam penggunaannya, jika karakter ’i’ pada pattern bertemu dengan karakter ’r’ pada teks, maka pattern bisa dipindahkan sejauh 1 karakter ke kanan. Begitu juga jika karakter ’c’ pada pattern kemudian bertemu dengan karakter ’y’ pada teks, maka pattern langsung dipindah sejauh 4 karakter ke kanan.

Setelah tabel dibentuk, barulah pencocokan string dilakukan . Proses pencocokan adalah dengan membandingkan karakter antara teks dan pattern. Jika menemui ketidakcocokan pada proses ini, maka pattern dipindahkan sejauh jumlah pada tabel yang bersangkutan. Proses ini dilakukan hingga string yang kita cari ditemukan atau karakter dalam teks mencapai akhir.

5. Kesimpulan

Algoritma Boyer-Moore cukup ampuh digunakan dalam pencarian string terutama pada teks. Algoritma ini merupakan perbaikan dari algoritma brute-force dan KMP. Algoritma ini melakukan pendekatan untuk tidak mengecek semua karakter pada teks dengan melompatinya, sehingga banyak karakter yang tidak perlu diperiksa sia-sia.

Akibatnya, waktu yang digunakan menjadi lebih optimal.

Namun, kekurangan juga tetap dimiliki oleh algoritma ini. Untuk teks atau data yang tersusun atas karakter *unicode*, pembuatan tabel akan semakin besar. Tentu saja ini akan membutuhkan memori yang cukup besar pula.

Referensi

[1]Kim, Jong Yong and John Shawe Taylor. 1994. *Fast String Matching using an n-gram Algorithm*. www.cs.ubc.ca/local/reading/proceedings/spe91-95/spe/vol24/issue1/spe872.pdf

[2]Michael T. Goodrick and Roberto Tamassia. 2002. *Algorithm Design*. John Wiley and Sons, Inc.

[3]Mohamad, Ababneh and Oqeili Saleh and Rawan Abdeen. 2006. *Occurences Algorithm for String Based on Brute-Force Algorithm* on Journal of Computer Science 2: 82. Science Publications.

[4]Munir, Rinaldi. *Strategi Algoritmik*, Lab Ilmu dan Rekayasa Komputasi, Departemen Teknik Informatika ITB. 2005

[5]National Institute of Standards and Technology. 2006. *String Matching*. http://www.nist.gov/dads/HTML/stringmatching.html.