PENCOCOKAN DNA PATTERN DENGAN ALGORITMA BOYER-MOORE

Satya Fajar Pratama

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
e-mail: satya_fajar@yahoo.co.uk

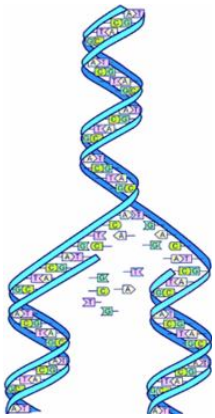
ABSTRAK

Pencocokan string adalah suatu masalah yang hampir terjadi di seluruh bidang kehidupan. Salah satu contohnya adalah pencocokan rangkaian DNA. Rangkaian DNA yang menyimpan cetak biru bagi segala aktivitas sel ini merupakan suatu bagian yang penting dalam tubuh manusia. Segala percobaan yang menyangkut DNA, termasuk pencocokan rangkaian DNA, diharapkan akan membawa kemajuan dalam dunia kedokteran di kemudian hari. Sebenarnya, terdapat banyak algoritma pencocokan string yang dapat diterapkan dalam pencocokan rangkaian DNA. Akan tetapi, penggunaan algoritma yang tepat dan mangkus dalam mencocokkan DNA yang berbeda akan sangat membantu para ilmuwan dalam mempelajari rangkaian DNA tersebut. Oleh karena itu, makalah ini akan membahas algoritma Boyer-Moore, sebuah algoritma pencocokan string yang mangkus dan sangkil, serta penerapannya dalam pencocokan DNA.

Kata kunci: Algoritma Boyer-Moore, Pencocokan DNA.

1. PENDAHULUAN

DNA merupakan polimer yang terdiri dari 3 komponen utama, yaitu gugus fosfat, gula deoksiribosa, dan basa nitrogen. DNA (asam deoksiribonukleat) yang terletak di dalam sebuah sel berperan sebagai materi genetik, yaitu penyimpan informasi untuk segala aktivitas sebuah sel. DNA tersebut berbentuk dua untaian yang berpilin (*double helix*). Masing-masing untaian terdiri dari rangka utama dan basa nitrogen yang berinteraksi dengan untaian DNA lainnya. Kedua untaian DNA tersebut akan disatukan oleh ikatan hidrogen antara basa-basa yang terdapat masing-masing untaian. Empat basa yang terdapat pada DNA adalah adenin (A), sitosin (C), guanin (G), dan timin (T).



Gambar 1. *Double-Helix* DNA

Karena peran dan fungsi DNA yang sangat krusial dalam tubuh manusia, banyak ilmuwan melakukan berbagai macam riset untuk memanipulasi rangkaian DNA. Rangkaian DNA tersebut akan sangat berpengaruh terhadap materi genetik yang diturunkan oleh sang induk kepada anaknya. Proses pencocokan DNA merupakan suatu hal yang hampir selalu digunakan dalam berbagai proses manipulasi rangkaian DNA. Pada umumnya, proses ini bertujuan untuk mendapatkan karakteristik dari suatu DNA. Oleh karena itu, diperlukan sebuah algoritma pencocokan DNA yang efisien untuk mengurangi kebutuhan waktu yang terjadi (kompleksitas waktu rendah) dan mendapatkan karakteristik DNA yang tepat.

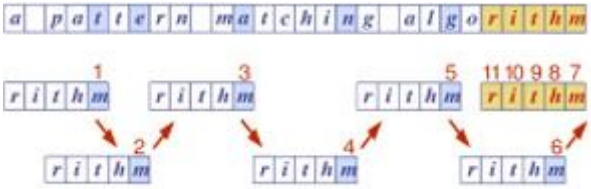
2. ALGORITMA BOYER-MOORE

2.1. Karakteristik Algoritma Boyer-Moore

Algoritma Boyer-Moore adalah algoritma pencarian string yang paling efektif saat ini. Algoritma yang ditemukan oleh Bob Boyer dan J. Strother Moore ini telah menjadi standar untuk berbagai literatur pencarian string. Algoritma Boyer-Moore akan menyimpan informasi pergeseran untuk melakukan pencarian string.

Karakteristik utama dari algoritma Boyer-Moore adalah algoritma ini melakukan pencocokan string mulai dari

kanan (belakang). Dengan karakteristik tersebut, ketidakcocokan saat terjadi perbandingan string akan membuat pergerakan *pattern* melompat lebih jauh untuk menghindari perbandingan karakter pada string yang diperkirakan gagal.



Gambar 2. Algoritma Boyer-Moore

2.2. Penjelasan Algoritma Boyer-Moore

Algoritma Boyer-Moore menggunakan dua buah tabel untuk mengolah informasi saat terjadi kegagalan pencocokan string. Tabel pertama akan menghitung banyaknya pergeseran yang harus dilakukan berdasarkan identitas karakter yang menyebabkan kegagalan pencocokan string. Jumlah pergeseran yang terdapat pada tabel pertama ini sering disebut dengan *bad character shift*. Tabel kedua juga akan menghitung banyaknya pergeseran yang harus dilakukan, tetapi berdasarkan jumlah karakter yang berhasil dicocokkan sebelum pencocokan string tersebut gagal. Jumlah pergeseran yang dihitung oleh tabel kedua ini dikenal dengan istilah *good suffix shift.*

2.3. Pseudo-code Algoritma Boyer-Moore

```
procedure preBmBc(input x: larik karakter, input
m: integer, input/output bmBc: larik integer)

Deklarasi
   i:integer

Algoritma
   for (i=0; i<ASIZE; ++i)
      bmBc[i] ← m
   for (i=0; i < m-1; ++i)
      bmBc[x[i]] ← m-i-1


procedure suffixes(input x: larik karakter, input
m: integer, input/output suff: larik integer)

Deklarasi
   f, g, i : integer

Algoritma
    suff[m-1] ← m;
    g ← m-1

   for (i = m-2; i >= 0; --i)
      if (i>g and suff[i+m-1-f]<i-g)
         suff[i] → suff[i+m-1-f]
      else
        if (i<g)
           g ← i
        f = i
        while(g> 0 and x[g]=x[g+m-1-f])
            --g
        suff[i] ← f-g

procedure preBmGs(input x: larik karakter, input
m:integer, input/output bmGs: larik integer)

Deklarasi
   i, j : integer
   suff : array [0..XSIZE] of integer
Algoritma
   suffixes(x, m, suff)

   for (i = 0; i < m; ++i)
      bmGs[i] ← m
   j ← 0
   for (i = m-1; i >= -1; --i)
      if (i = -1 or suff[i] = i+1)
         for (j = 0; j < m-1-i; ++j)
            if (bmGs[j] = m)
               bmGs[j] ← m-1-i
   for (i = 0; i <= m-2; ++i)
      bmGs[m-1-suff[i]] ← m-1-i

procedure BM(input x: larik karakter, input
m:integer, input y: larik karakter, input
n:integer)

Deklarasi
   i, j : integer
   bmGs : array [0..XSIZE] of integer
   bmBc : array [0..ASIZE] of integer

Algoritma
   /* Preprocessing */
   preBmGs(x, m, bmGs)
   preBmBc(x, m, bmBc)

   /* Searching */
   j = 0
   while (j <= n-m)
      for (i = m-1; i >= 0 and x[i]= y[i+j]; --i)
        if (i < 0)
          OUTPUT(j)
          j ← j + bmGs[0]
        else
          j ← j + MAX(bmGs[i], bmBc[y[i+j]]-
m+1+i)
```

3. IMPLEMENTASI ALGORITMA BOYER-MOORE DALAM PENCOCOKAN DNA PATTERN

Misalkan pada suatu rangkaian DNA (S)
GCATCGCAGAGAGTATACAGTACG

akan dicari suatu pola (*pattern*) DNA (P)
GTATAC

Sebelum memulai proses pencocokan rangkaian DNA tersebut, terlebih dahulu harus dibuat 2 buah tabel yang akan menyimpan informasi pergeseran, seperti yang telah dijelaskan sebelumnya.

a	c	g	t
1	6	5	2

Tabel 1 bmBc

0	1	2	3	4	5
6	6	6	6	6	1

Tabel 2 BmGs

Proses-proses yang terjadi pada pencocokan rangkaian DNA tersebut sebagai berikut :

Attempt 1
GCATC**G**CAGAGAGTATACAGTACG
GTATA**C**

(bmBc[g]-6+6)

Karakter paling kanan pada *pattern* P , yaitu C, akan dicocokkan dengan karakter yang berada sejajar dengannya, yaitu G. Karena kedua karakter tersebut berbeda, maka *pattern* P akan digeser sejauh 5 karakter atau sampai karakter G pada *pattern* P berada sejajar dengan karakter G yang terdapat pada S.

Attempt 2
GCATCGCAGA**G**AGTATACAGTACG
GTATA**C**

(bmBc[g]-6+6)

Karakter terakhir pada *pattern* P, yaitu C, akan dibandingkan dengan karakter G yang terdapat pada S. Karena pencocokan karakter ini gagal, maka *pattern* P akan digeser kembali sedemikian sehingga karakter G pada *pattern* P sejajar dengan karakter G pada S.

Attempt 3
GCATCGCAGAGAGTA**T**ACAGTACG
GTATA**C**

(bmBc[t]-6+6)

Karakter terakhir pada *pattern* P, yaitu C, akan dibandingkan dengan karakter yang sejajar dengannya, yaitu T. Karena kedua karakter tersebut berbeda, maka *pattern* P akan digeser sedemikian rupa sehingga karakter T terakhir yang terdapat pada *pattern* P berada sejajar dengan karakter T pada S.

Attempt 4
GCATCGCAGAGA**GTATAC**AGTACG
GTATAC

(bmGs[0])

Karakter terakhir pada *pattern* P akan dibandingkan dengan karakter yang berada sejajar dengannya. Karena perbandingan kedua karakter ini berhasil, maka pencocokan karakter dilanjutkan terhadap karakter sebelumnya. Hal ini akan terus dilakukan sampai semua karakter pada *pattern* P telah dibandingkan. Pada *attempt* (tahapan) ini, *pattern* DNA P telah ditemukan. Akan tetapi, karena S belum habis, maka pencocokan tetap akan dilanjutkan. *Pattern* P akan digeser sedemikian sehingga karakter terakhir pada *pattern* P berada sejajar dengan karakter yang sama pada S.

Attempt 5
GCATCGCAGAGAGTATACA**GTAC**G
GT**ATAC**

(bmBc[g]-6+6)

Karakter terakhir pada *pattern* P, yaitu C, akan dibandingkan dengan karakter yang berada sejajar dengannya. Karena kedua karakter tersebut sama, maka perbandingan dilanjutkan terhadap karakter sebelumnya. Hal ini dilakukan sampai pencocokan karakter tidak berhasil, yaitu saat perbandingan antara karakter A pada *pattern* P dan karakter G pada S terjadi. Selanjutnya, *pattern* P akan digeser sejauh 5 karakter. Karena karakter paling kanan pada *pattern* P tidak memiliki pasangan yang berada sejajar dengannya, maka pencarian *pattern* DNA P dihentikan.

4. KESIMPULAN

Algoritma Boyer-Moore memiliki keunggulan dari algoritma pencocokan string lainnya. Keunggulan yang pertama adalah algoritma Boyer-Moore melakukan pencocokan string dari kanan (belakang). Keunggulan berikutnya adalah algoritma Boyer-Moore menggunakan dua buah tabel untuk menyimpan informasi pergeseran, yaitu *bad character shift* dan *good suffix shift*, sehingga kompleksitas waktu yang dibutuhkan tergolong rendah.

Karena keunggulan-keunggulan tersebut, algoritma Boyer-Moore merupakan algoritma pencocokan string yang paling efektif saat ini sehingga sangat tepat diimplementasikan untuk melakukan pencarian *pattern* DNA pada suatu rangkaian DNA tertentu.

REFERENSI

[1] Jorma Tarhio and Hannu Peltola ,"String Matching in the DNA Alphabet", *Software-Practice and Experience*, 1997
[2] http://www.cs.aau.dk/~simas/aalg04/slides/aalg3.ppt
[3] http://id.wikipedia.org
[4] http://www-igm.univ-mlv.fr