Diagnosa *Sickle-Cell Anemia* dengan Pencarian String

Atika Firdaus / 13514009
Program Studi Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13514009@std.stei.itb.ac.id

***Abstract*—Dalam ilmu komputasional, terdapat persoalan-persoalan tertentu dimana dibutuhkan pencocokan antara satu *pattern* dengan *pattern* yang lain. Dari sana lah tercipta algoritma-algoritma pencarian string. Beberapa algoritma pencarian string yang umum dipakai adalah *Brute-Force* dan Knuth-Moris-Pratt. Prinsip-prinsip dalam algoritma pencarian string dapat digunakan untuk menyelesaikan isu-isu di bidang lain, salah satunya proses analisis struktur kodon DNA untuk menemukan kelainan genetik. Makalah ini akan membahas penggunaan pencarian string dengan algoritma *Brute-Force* dan Knuth-Moris-Pratt dalam mendeteksi penyakit *Sickle-Cell Anemia*.**

***Keywords*—Boyer Moore, *Brute-Force*, DNA, genetik, Knuth-Morris-Pratt, pencarian string.**

I. PENDAHULUAN

Salah satu hal penting dari suatu organisme adalah DNA. *Deoxyribonucleic Acid*/Asam Deoksiribonukleat/DNA merupakan sejenis biomolekul yang terdapat pada setiap organisme dan banyak jenis virus. DNA menyimpan informasi genetik dari organisme/virus berkaitan. Informasi genetik ini menentukan sifat (fenotip) dan karakteristik metabolisme dari pemiliknya. Selain itu, informasi genetik juga dapat diwariskan kepada keturunan.

Beberapa bagian DNA manusia memiliki struktur yang sama dengan manusia lainnya. Salah satunya adalah hemoglobin. Hemoglobin merupakan komponen pembentuk sel darah merah yang berfungsi mengangkut oksigen selama peredaran darah. Peranannya yang penting membuat sedikit saja perubahan pada struktur hemoglobin dapat berakibat fatal.

Salah satu kelainan pada hemoglobin manusia adalah *Sickle-Cell Anemia* atau dalam bahasa Indonesia disebut sebagai Anemia Sel Bulan Sabit. Kelainan ini menyebabkan bentuk fisik sel darah merah yang semula lingkaran dengan cekungan di tengah berubah menjadi berbentuk seperti bulan sabit. Bentuk fisiknya yang berubah menyebabkan fungsionalitas sel darah merah sebagai pengangkut oksigen terganggu. Selain itu, fleksibilitas sel darah merah menurun dan dapat memicu tersangkutnya sel darah merah selama perjalanannya dalam pembuluh darah.

Terdapat beberapa cara untuk mendeteksi apakah seseorang mengidap *Sickle-Cell Anemia*. Salah satu caranya adalah memeriksa sekuens asam amino yang tersusun dalam DNA hemoglobin. Karakteristik yang eksak dari DNA hemoglobin penderita *Sickle-Cell Anemia* mempermudah diagnosa terhadap pasien. Prinsip dari algoritma pencarian string dapat dimanfaatkan dalam pemeriksaan DNA ini.

II. LANDASAN TEORI

A. Konsep String

String merupakan tipe data yang pada dasarnya merupakan kumpulan karakter-karakter yang tersusun di dalam *array*. Pemilihan string sebagai bentuk penyimpanan kumpulan karakter memiliki banyak kelebihan dibanding penyimpanan dalam bentuk *array of character*. Banyak bahasa pemrograman yang menyediakan *method-method* dari tipe data string yang memudahkan penggunanya dalam melakukan operasi terhadap string berkaitan.

Misalkan, S adalah sebuah string berukuran m

$S = x_1x_2x_3x_4 \ldots x_m$

prefiks dari S adalah substring S[1..k-1] dan sufiks dari S adalah substring S[k-1..m] dengan k adalah sembarang indeks antara 1 dan m dan S[0] adalah karakter *null* yang disimbolkan sebagai ∅.

Contoh:

S = andrew

Seluruh kemungkinan prefiks dari S adalah ∅, a, an, and, andr, andre dan seluruh kemungkinan sufiks dari S adalah ∅, w, ew, rew, drew, ndrew.

B. Definisi Pencarian String

Pencarian string di dalam teks disebut juga sebagai pencocokan string, atau dalam bahasa Inggris umumnya disebut sebagai *string matching* atau *pattern matching.* Persoalan pencarian string dirumuskan sebagai berikut:

Diberikan:

1. Teks (*text*), yaitu string yang berukuran n karakter
2. *Pattern*, yaitu string yang berukuran m karakter, dengan syarat m < n, yang akan dicari di dalam teks

Tugas yang harus dilakukan adalah mencari (*find* atau *locate*) lokasi (indeks) pertama di dalam teks yang

bersesuaian dengan *pattern*. Jika *pattern* tidak ada pada teks, umumnya program akan mengembalikan indeks sebagai penanda tidak adanya *pattern*, -1 misalnya. Persoalan pencarian string banyak diaplikasikan dalam perangkat lunak, salah satu contohnya adalah fitur pencarian kata dalam dokumen (Find dalam Microsoft Office Word).

Contoh, dengan mengasumsikan spasi terhitung sebagai karakter dalam string:

1. *Pattern*: hari
   Teks: kami pulang **hari** kamis
   Solusi dari contoh ini adalah indeks 13
2. *Pattern*: not
   Teks: nobody **not**iced him
   Solusi dari contoh ini adalah indeks 8
3. *Pattern*: apa
   Teks: Si**apa** yang menjemput Papa dari kota Balikpapan?
   Solusi dari contoh ini adalah indeks 3

Terdapat beberapa algoritma yang digunakan dalam melakukan pencarian string, di antaranya adalah algoritma *Brute-Force* dan Knuth-Morris-Pratt. Perbedaan dari ketiga algoritma ini cukup signifikan, dilihat dari prinsip kerja yang digunakan serta kompleksitas waktu rata-rata yang dibutuhkan untuk menyelesaikan sebuah kasus pencarian string.

C. Algoritma *Brute-Force*

Misal, terdapat teks dengan ukuran n karakter yang tersimpan dalam array T[1..n] dan *pattern* dengan ukuran m karakter yang tersimpan dalam array P[1..m], maka cara kerja algoritma *Brute-Force* dalam pencarian string adalah:

1. *Pattern* P dicocokkan pada awal teks T.
2. Pencocokan bergerak dari arah kiri ke kanan, maka selanjutnya dilakukan pembandingan setiap karakter di dalam *pattern* P dengan karakter yang bersesuaian di dalam teks T hingga:
   a. semua karakter pada *pattern* P yang dibandingkan sama, yang berarti pencarian berhasil, atau
   b. terdapat ketidakcocokan karakter, yang berarti pencarian belum berhasil
3. Jika belum ditemukan kecocokan antara *pattern* P dengan teks T dan masih ada bagian teks T yang belum dianalisis, maka geser *pattern* P sebanyak satu karakter ke kanan. Proses selanjutnya adalah mengulang tahap nomor 2.

Contoh:

1. Teks: nobody noticed him
   Pattern: not

```
          nobody noticed him
s=0       not
s=1        not
s=2         not
s=3          not
s=4           not
s=5            not
s=6             not
s=7              not
```

2. Teks: 10010101001011110101010001

```
Pattern: 001011
         10010101001011110101010001
s=0      001011
s=1       001011
s=2        001011
s=3         001011
s=4          001011
s=5           001011
s=6            001011
s=7             001011
s=8              001011
```

Pseudocode algoritma *Brute-Force*:

```
procedure BruteForceSearch(input m, n : integer,
input P : array[1..m] of char, input T : array[1..n] of
char, output idx : integer)

Deklarasi
    s, j : integer
    ketemu : boolean

Algoritma
    s←0
    ketemu←false
    while (s ≤ n-m) and (not ketemu) do
        j←1
        while (j ≤ m) and (P[j] = T[s+j]) do
            j←j+1
        endwhile { j > m or P[j] ≠ T[s+j] }
        if j = m then { kecocokan string ditemukan }
            ketemu←true
        else
            { geser pattern satu karakter ke kanan
            teks }
            s←s+1
        endif
    endfor
    { s > n – m or ketemu }
    if ketemu then
        idx←s+1
    else
        idx←-1
    endif
```

Terdapat tiga jenis kompleksitas waktu dari algoritma ini, yaitu:

1. Kasus terburuk (*worst case*) membutuhkan jumlah perbandingan sebanyak m(n – m + 1), yang berarti kompleksitasnya adalah O(mn). Contoh:
   Teks: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaab
   Pattern: aaaab
2. Kasus terbaik (*best case*) memiliki kompleksitas O(n). Kasus terbaik terjadi ketika karakter pertama *pattern* tidak pernah sama dengan karakter teks yang sedang dicocokkan. Oleh karena itu, jumlah perbandingan yang dilakukan paling banyak n kali. Contoh:
   Teks: Ini adalah string yang berakhir dengan zz
   Pattern: zz

3. Kasus rata-rata (*average case*) memiliki kompleksitas O(m+n). Contoh:
   Teks: a string searching example is standard
   Pattern: "store"

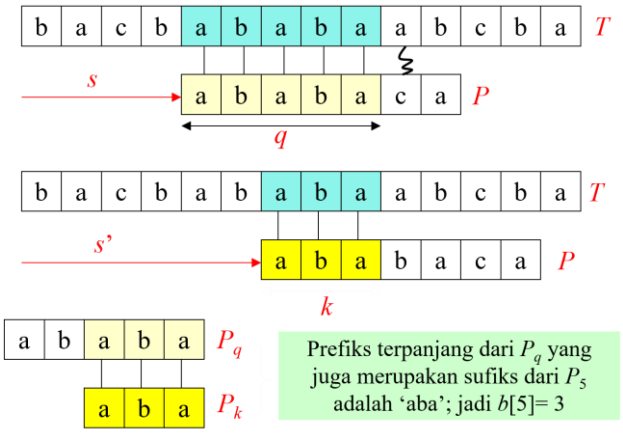
Algoritma *Brute-Force* cenderung cepat ketika alphabet dari teks berjumlah banyak, seperti A..Z, a..z, 0..9, dll. Dalam kondisi sebaliknya, algoritma ini cenderung lambat, seperti 0, 1 (pada *file binary*, gambar, dll.).

D. Algoritma Knuth-Morris-Pratt

Asal mula nama algoritma Knuth-Morris-Pratt, singkatnya KMP, adalah dari para pengembangnya, yaitu D. E. Knuth, J. H. Morris, dan V. R. Pratt. Algoritma KMP melakukan pencocokan *pattern* dengan teks dari arah kiri ke kanan, sama seperti algoritma *Brute-Force*. Tetapi, hal yang membedakan algoritma KMP dengan *Brute-Force* adalah proses penggeseran yag dilakukan ketika ditemukan ketidakcocokan antara *pattern* dengan teks.

Pada algoritma *Brute-Force*, ketika ditemukan ketidakcocokan, maka *pattern* akan digeser satu karakter ke kanan. Lain halnya dengan algoritma KMP, algoritma ini menyimpan informasi yang kemudian dimanfaatkan untuk menghitung jumlah pergeseran yang harus dilakukan ketika ditemukan ketidakcocokan. Dengan prinsip ini, waktu pencarian dapat berkurang secara signifikan.

Misal, ketidakcocokan terjadi pada posisi *pattern* P[j], maka pergeseran yang harus dilakukan untuk menghindari perbandingan yang boros adalah sebanyak jumlah maksimal prefiks dari P[1..j-1] yang juga merupakan sufiks dari P[1..j-1].



Pada algoritma KMP, dilakukan pemrosesan awal terhadap *pattern* untuk menemukan kecocokan antara prefiks dan *pattern* itu sendiri. Pemrosesan awal ini disebut sebagai *border function*/fungsi pinggiran dan memiliki nama lain *failure function*. Ketika terjadi ketidakcocokan antara *pattern* dengan teks pada P[j], maka akan dilakukan pemeriksaan terhadap fungsi pinggiran dari k=j-1.

Sebagai contoh, misal terdapat *pattern* P = ababaa. Nilai fungsi pinggiran untuk setiap karakter di dalam P adalah:

j	1	2	3	4	5	6
P[j]	a	b	a	b	a	a
b(j)	0	0	1	2	3	1

Pseudocode Fungsi Pinggiran KMP

```
procedure HitungPinggiran(input m : integer, P :
array[1..m] of char, output b : array[1..m] of integer)

Deklarasi
    k,q : integer

Algoritma
    b[1]←0
    q←2
    k←0
    for q←2 to m do
        while ((k > 0) and (P[q] ≠ P[k+1])) do
            k←b[k]
        endwhile
        if P[q]=P[k+1] then
            k←k+1
        endif
        b[q]=k
    endfor
```

Pseudocode Algoritma KMP

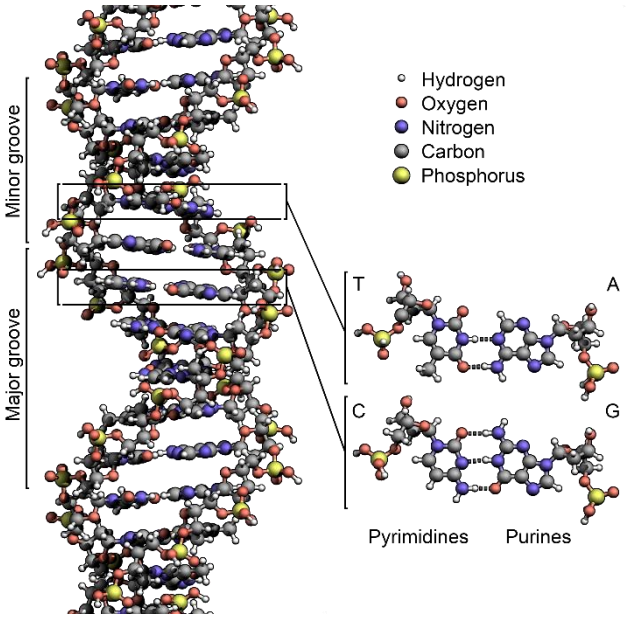
```
procedure KMPsearch(input m, n : integer, input P :
array[1..m] of char, input T : array[1..n] of char,
output idx : integer)

Deklarasi
    i, j : integer
    ketemu : boolean
    b : array[1..m] of integer
    procedure HitungPinggiran(input m : integer, P :
    array[1..m] of char, output b : array[1..m] of
    integer)

Algoritma
    HitungPinggiran(m, P, b)
    j←0
    i←1
    ketemu←false
    while (i ≤ n and not ketemu) do
        while((j > 0) and (P[j+1]≠T[i])) do
            j←b[j]
        endwhile
        if P[j+1]=T[i] then
            j←j+1
        endif
        if j = m then
            ketemu←true
        else
            i←i+1
        endif
    endwhile
    if ketemu then
        idx←i-m+1
    else
        idx←-1
    endif
```

E. DNA (*Deoxyribonucleic Acid)*

Asam deoksiribonukleat, yang lebih dikenal dari singkatannya yaitu DNA. Merupakan sejenis biomolekul yang menyimpan informasi genetika dari setiap organisme dan banyak jenis virus. Informasi-informasi genetika ini memiliki peran penting dalam pertumbuhan, perkembangan, dan fungsi dari organisme dan virus. DNA merupakan asam nukleat.

Asam nukleat sendiri adalah makromolekul esensial bagi seluruh makhluk hidup yang diketahui. Kebanyakan molekul DNA terdiri dari dua untai biopolimer yang berlilitan satu sama lain membentuk *double-helix*. Setiap untai biopolimer tersusun dari satuan-satuan molekul yang disebut nukleotida. Oleh karena itu, untaian ini dikenal sebagai polinukleotida. Tiap nukleotida terdiri atas gula monosakarida yang disebut deoksiribosa, gugus fosfat, dan salah satu jenis basa nitrogen, yaitu guanin (G), adenin (A), timin (T), atau sitosin (C). Kumpulan nukleotida ini kemudian tersambung dalam satu rantai ikatan kovalen antara deoksiribosa satu nukleotida dengan gugus fosfat nukleotida yang lain. Ikatan ini menghasilkan rantai gula-fosfat yang berselang-seling. Menurut aturan pasangan basa nitrogen, yaitu A dengan T dan C dengan G, ikatan hidrogen mengikat basa-basa dari kedua untai polinukleotida membentuk DNA *double-helix*.

Dua untai DNA bersifat anti-paralel, yaitu keduanya berpasangan secara berlawanan. Pada setiap gugus gula, terikat salah satu dari empat jenis basa nitrogen. Urutan-urutan empat basa nitrogen di sepanjang rantai DNA inilah yang menyimpan kode informasi biologis. Struktur kimia DNA yang ada membuatnya sangat cocok untuk menyimpan informasi biologis setiap makhluk hidup.



Sumber:
https://en.wikipedia.org/wiki/File:DNA_Structure%2BKey%2BLabelled.pn_NoBB.png

nonpolar polar basic acidic (stop codon)

Standard genetic code

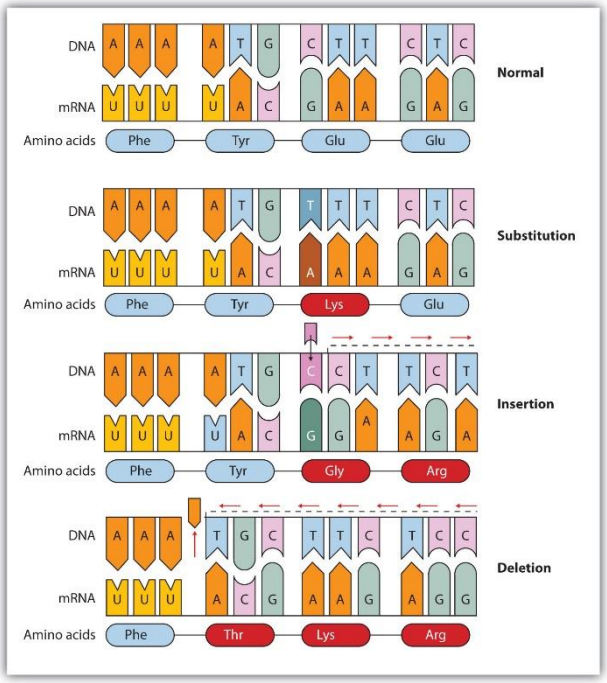
1st base	2nd base T		2nd base C		2nd base A		2nd base G		3rd base
T	TTT	(Phe/F) Phenylalanine	TCT	(Ser/S) Serine	TAT	(Tyr/Y) Tyrosine	TGT	(Cys/C) Cysteine	T
	TTC		TCC		TAC		TGC		C
	TTA	(Leu/L) Leucine	TCA		TAA	Stop (*Ochre*)	TGA	Stop (*Opal*)	A
	TTG		TCG		TAG	Stop (*Amber*)	TGG	(Trp/W) Tryptophan	G
C	CTT		CCT	(Pro/P) Proline	CAT	(His/H) Histidine	CGT	(Arg/R) Arginine	T
	CTC		CCC		CAC		CGC		C
	CTA		CCA		CAA	(Gln/Q) Glutamine	CGA		A
	CTG		CCG		CAG		CGG		G
A	ATT	(Ile/I) Isoleucine	ACT	(Thr/T) Threonine	AAT	(Asn/N) Asparagine	AGT	(Ser/S) Serine	T
	ATC		ACC		AAC		AGC		C
	ATA		ACA		AAA	(Lys/K) Lysine	AGA	(Arg/R) Arginine	A
	ATG[A]	(Met/M) Methionine	ACG		AAG		AGG		G
G	GTT	(Val/V) Valine	GCT	(Ala/A) Alanine	GAT	(Asp/D) Aspartic acid	GGT	(Gly/G) Glycine	T
	GTC		GCC		GAC		GGC		C
	GTA		GCA		GAA	(Glu/E) Glutamic acid	GGA		A
	GTG		GCG		GAG		GGG		G

Sumber:
https://en.wikipedia.org/wiki/DNA_codon_table

F. Mutasi DNA

Mutasi berasal dari sebuah kata dalam bahasa Latin *Mutatus* yang berarti perubahan. Mutasi dalam biologi didefinisikan sebagai perubahan permanen urutan nukleotida pada DNA dari suatu organisme, virus, maupun elemen genetik lainnya yang dapat diwariskan secara genetik kepada keturununannya. Mutasi dapat terjadi karena kerusakan DNA yang tidak terperbaiki, kesalahan pada proses replikasi DNA, atau dari insersi/delesi segmen DNA yang dilakukan oleh *mobile genetic elements* (MGE).

Penyebab terjadinya mutasi, disebut sebagai mutagen, dikelompokkan menjadi tiga hal yaitu kimiawi, fisik, dan biologis. Mutasi dapat terjadi baik di tingkat gen maupun kromosom. Mutasi gen, yang kemudian disebut juga sebagai mutasi titik, merupakan sebuah mutasi yang terjadi karena perubahan satu pasang basa nitrogen pada DNA. Perubahan ini mempengaruhi asam amino pada protein yang dibentuk, yang kemudian menyebabkan perubahan metabolisme dan fenotip (sifat) dari suatu organisme.



Sumber:
http://2012books.lardbucket.org/books/introduction-to-chemistry-general-organic-and-biological/section_22/4b82e479bd31db665696203cea437b72.jpg

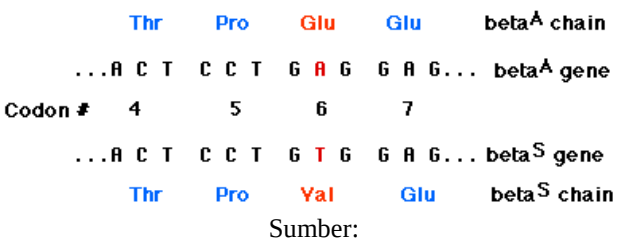
G. *Sickle-Cell Anemia*

Sickle-Cell Anemia yang dalam bahasa Indonesia disebut sebagai Anemia Sel Bulan Sabit merupakan salah satu contoh dari penyakit yang dilatarbelakangi oleh mutasi DNA. Penyakit ini mengganggu proses peredaran darah dan dikategorikan sebagai penyakit yang sangat membahayakan penderitanya. Sebagaimana penyakit kelainan DNA yang lain, penyakit ini dapat diwariskan kepada keturunan sang penderita.

Bagian spesifik yang terpengaruhi mutasi DNA pada penyakit *Sickle-Cell Anemia* adalah hemoglobin atau sel darah merah. Hemoglobin yang terdapat dalam tubuh manusia dewasa pada umumnya disebut sebagai hemoglobin A (HbA). Namun, pada penderita *Sickle-Cell Anemia*, terdapat bentuk lain dari hemoglobin yang terdapat dalam tubuh mereka. Jenis hemoglobin ini disebut hemoglobin S (HbS).

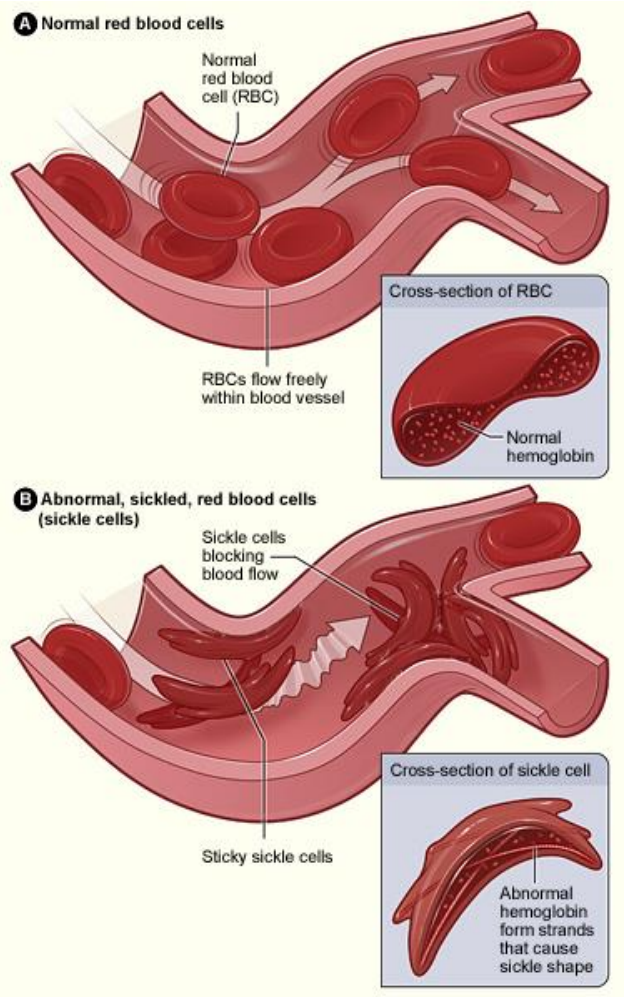
Nama *Sickle-Cell Anemia*/Anemia Sel Bulan Sabit didapat dari bentuk sel darah merah penderitanya yang berubah bentuk menjadi seperti bulan sabit ketika melewati pembuluh darah terutama pada jaringan yang sedang bermetabolisme aktif, di mana sel darah merah normal berbentuk lingkaran dengan bagian pipih di tengahnya. Bentuk bulan sabit ini menyebabkan sel darah merah menjadi tidak sefleksibel sel darah merah dalam bentuk normalnya. Selain itu, sel darah merah penderita *Sickle-Cell Anemia* cenderung lebih lemah dan memiliki waktu hidup yang jauh lebih singkat. Hal inilah yang kemudian menyebabkan sang penderita penyakit mengalami anemia karena kekurangan sel darah merah.

Hemoglobin merupakan protein yang ditemukan pada sel darah merah. Hemoglobin terdiri dari dua subunit, yaitu α-globin dan β-globin. Pada penderita *Sickle-Cell Anemia*, terjadi mutasi titik pada β-globin yang menyebabkan asam amino glutamat yang bersifat hidrofilik digantikan posisinya dengan asam amino valin yang bersifat hidrofobik pada posisi keenam (tidak termasuk *start codon*). Gen β-globin terletak pada kromosom 11. Jika terdapat dua α-globin subunit dan dua mutan β-globin, maka terbentuklah HbS.



Sumber:
http://www.biology-pages.info/S/SickleMutation.gif

Perubahan terhadap satu saja dari 146 asam amino dapat menyebabkan perbedaan yang begitu signifikan karena sifat yang sangat jauh berbeda antara glutamat dengan valin. Glutamat merupakan molekul hidrofilik yang kuat, sedangkan valin merupakan molekul hidrofobik yang kuat. Posisi asam amino dimana terjadi mutasi terletak pada permukaan rantai β-globin yang mana berhadapan langsung dengan air. Hal ini kemudian mengurangi kelarutan molekul dan memicu pembentukan gumpalan.



Sumber:
https://upload.wikimedia.org/wikipedia/commons/a/ac/Sickle_cell_01.jpg

III. MENDETEKSI CIRI *SICKLE-CELL ANEMIA* PADA DNA MENGGUNAKAN PENCARIAN STRING

Perbedaan antara penderita *Sickle-Cell Anemia* dan orang yang sehat terletak pada sekuens asam amino pada hemoglobinnya, lebih spesifiknya pada gen β-globin. Pada manusia, gen ini terletak pada kromosom ke-11. Maka, untuk melakukan diagnosa terhadap seseorang apakah mengidap *Sickle-Cell Anemia* dapat dilakukan dengan melakukan pemeriksaan sekuens asam amino yang tersusun di dalam gen β-globin pada kromosom ke-11.

Posisi eksak dari lokasi terjadinya mutasi gen memudahkan diagnosa yang dilakukan. Perbedaan pasti terjadi pada gen β-globin asam amino keenam. Maka,

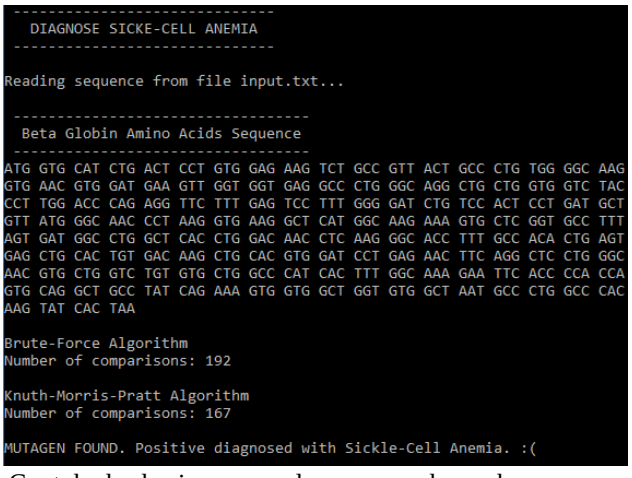
pencarian string dapat dilakukan dengan menggunakan valin, GTG, sebagai *pattern* yang akan dicari. Sedangkan, teks yang akan diuji merupakan sekuens asam amino dari gen β-globin manusia.

Sekuens asam amino dari gen β-globin manusia memiliki panjang 441 karakter. Angka ini didapat dari jumlah total asam amino dari gen β-globin pada manusia normal, ditambah satu asam amino sebagai *start codon*, dan satu asam amino sebagai *termination codon*, yang kemudian dikalikan dengan tiga karena setiap satu asam amino tersusun atas tiga basa nitrogen. Sedangkan, *pattern* memiliki panjang tiga karakter.

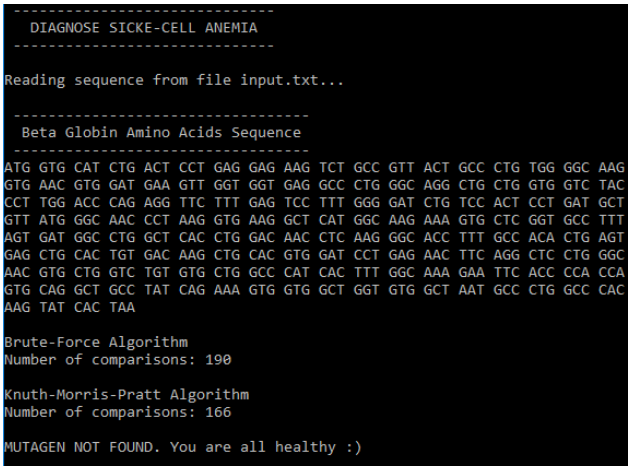
Kedua algoritma string yang akan digunakan dalam analisis ini melakukan pencarian terhadap posisi pertama kali *pattern* ditemukan di dalam teks, dan mengembalikan indeks/posisi. Tetapi, pada sekuens gen β-globin manusia terdapat valin yang terletak sebelum kodon keenam. Oleh karena itu, algoritma pencarian string yang mulanya mengembalikan indeks pertama ditemukannya *pattern*, diubah menjadi mengembalikan sebuah *array* yang menyimpan indeks-indeks dimana saja *pattern* ditemukan pada teks. Kemudian, dilakukan pemeriksaan apakah dalam *array* tersebut terdapat indeks 19 (dengan asumsi teks dimulai dari indeks 1), yang berarti ada valin pada asam amino keenam. Jika ya, berarti pemilik DNA mengidap penyakit *Sickle-Cell Anemia*.

IV. UJI COBA MENGGUNAKAN PROTOTIPE

Dalam prototipe yang saya buat, program menerima masukan berupa sekuens gen β-globin dari file eksternal. Kemudian, dilakukan pencarian string dengan algoritma *Brute-Force* dan KMP. Teks yang digunakan berasal dari masukan file eksternal, sedangkan *pattern* yang akan dicari merupakan konstanta berupa string “GTG”, asam amino valin. Program akan menampilkan ke layar sekuens gen β-globin yang dianalisis, kemudian memberi keluaran berupa jumlah perbandingan yang dilakukan oleh masing-masing algoritma serta hasil diagnosa *Sickle-Cell Anemia*.



Contoh eksekusi program dengan masukan sekuens asam amino gen β-globin dari penderita *Sickle-Cell Anemia*



Contoh eksekusi program dengan masukan sekuens asam amino gen β-globin dari non-penderita *Sickle-Cell Anemia*

V. KESIMPULAN

Algoritma pencarian string dapat diaplikasikan dalam banyak sekali hal yang berkaitan dengan kehidupan. Salah satunya adalah dalam mendiagnosa apakah seseorang menderita *Sickle-Cell Anemia*, sebuah penyakit yang disebabkan oleh mutasi DNA yang mempengaruhi bentuk sel darah merah dalam tubuh penderita. Perbedaan antara penderita dengan non-penderita *Sickle-Cell Anemia* dapat dideteksi dari sekuens asam amino pada gen β-globin tubuhnya. β-globin merupakan salah satu pembentuk hemoglobin, hemoglobin sendiri adalah komponen penyusun sel darah merah.

Dengan pendeteksian dini, penderita dan orang-orang terdekat penderita berkesempatan melakukan penanganan yang dapat meminimalisasi efek dari penyakit yang diderita.

VI. UCAPAN TERIMA KASIH

Puji syukur kehadirat Allah SWT, karena atas limpahan rahmat-Nya penulis dapat menyelesaikan makalah ini tepat waktu. Penulis mengucapkan terima kasih kepada dosen pengajar mata kuliah IF2211 Strategi Algoritma, Dr. Ir. Rinaldi Munir, M.T. dan Dr. Nur Ulfa Maulidevi, S.T., M.T. yang telah memberikan ilmunya kepada saya dan teman-teman mengenai Strategi Algoritma yang menjadi dasar pengetahuan dalam pembuatan tugas makalah ini. Dan kepada orang-orang di sekitar saya yang tidak dapat saya sebutkan satu-persatu, terima kasih atas dukungan yang diberikan kepada saya selama proses pembuatan makalah ini.

REFERENSI

[1] DNA-mutations. http://www.chemguide.co.uk/organicprops/aminoacids/dna6.html diakses pada tanggal 6 Mei 2016 pukul 12.07 WIB.

[2] Mutasi DNA. https://kamriantiramli.wordpress.com/tag/mutasi-dna/ diakses pada tanggal 8 Mei 2016 pukul 01.51 WIB.
[3] Mutasi Gen dan Mutasi Kromosom. http://www.cpuik.com/2013/05/mutasi-gen-dan-mutasi-kromosom.html diakses pada tanggal 8 Mei 2016 pukul 01.47 WIB.
[4] Russell, Peter. 2001. *iGenetics*. New York: Benjamin Cummings.
[5] Saenger, Wolfram. 1984. *Principles of Nucleic Acid Structure*. New York: Springer-Verlag.
[6] Sickle-Cell Disease. http://www.biology-pages.info/S/Sickle-cellDisease.html diakses pada tanggal 6 Mei 2016 pukul 12.03 WIB.

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, 8 Mei 2016

Atika Firdaus
13514009