# Bioinformatik Local Sequence Alignment dengan Algoritma Smith-Waterman dan Global Sequence Alignment dengan Algoritma Needlemen-Wunsch Serta Perbandingan antara kedua algoritma tersebut

Akbar Idria
Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Al Azhar Indonesia

***Abstract*—Algoritma *Smith-Waterman* dan Algoritma *Needleman-Wunsch* adalah algoritma *Dynamic Programming*. dimana dengan pendekatan *Dynamic Programming*, hasil dari *Sequence Alignment* antara urutan DNA, RNA atau Protein akan selalu optimal. jadi 2 algoritma yang akan saya gunakan memiliki tujuan yang sama yaitu untuk mengukur kemiripan dari urutan DNA yang satu dengan Urutan DNA yang lainnya.**

**Kata Kunci— Algoritma *Smith-Waterman*; Algoritma *Needleman-Wunsch*; *Dynamic Programming*; DNA; RNA; Protein;**

## I. PENDAHULUAN

Dalam bidang *Bioinformatics, Alignment* atau kesejajaran adalah salah satu teknik untuk mengidentifikasi atau menganalisis sekuens dari DNA, RNA ataupun protein. analisis ini nantinya akan dikaitkan dengan analasis evolusi dan analis lainnya yang membutuhkan kesejajaran dari DNA, RNA ataupun protein.

yang membuat menarik dari *Bioinformatic Sequence Alignment* adalah sangat banyak sekali kesuksesan dari penelitian *bioinformatic* bergantung pada s*equence alignment*, baik itu untuk DNA, RNA ataupun Protein. sehingga dengan optimalnya s*equence alignment* akan mempermudah dari penelitian-penelitian *bioinformatic*.

## II. DASAR TEORI

### *A. Sequence Alignment*

Dalam Bioinformatik, *Sequence Alignment* adalah cara untuk mengurutkan urutan-urutan DNA, RNA, Protein untuk mengidentifikasi daerah kesamaan diantara mereka. *Sequence Alignment* sendiri digunakan untuk menyimpulkan hubungan-hubungan funsional, struktural dan evolusi antara urutan tersebut [1].

### *B. Algoritma Smith-Waterman*

1. Pengertian

Algoritma *Smith-Waterman* dilakukan untuk local *sequence Alignment*. Yang bertujuan untuk menentukan daerah yang sama antara dua urutan, dan algoritma ini hanya membandingkan segmen-segmen yang memiliki kemiripan dari dua urutan yang ada [2]

2. Pseudo Code

*Pseudo code* untuk algoritma *Smith-Waterman* adalah [5] :

```
Input: 2 urutan String X dan Y
Output: lokal alignment dan nilai a
Inisialisasi:
Set M(i, 0) := 0 untuk semua
i = 0, 1, 2, . . ., n
Set M(0, j) := 0 untuk semua
j = 0, 1, 2, . . ., n
For i = 1, 2, . . . , n
do:
    For j = 1, 2, . . . , n
    do:
          Set M(i, j) = Max[0, M(i – 1, j – 1) + s(xi, yj ), M(i – 1, j) + w, M(i, j – 1) + w]
          Set backtrace T(i, j) to the maximizing pair (i` , j`) Set (i, j) := arg max{M(i, j) | i = 1, 2, . . . , n, j = 1, 2, . . . , m} The best score is α := M(i, j)
repeat
if T(i, j) = (i – 1, j – 1)
    print( xi-1, yj-1)
else if T(i, j) = (i – 1, j)
    print (xi-1, -)
else
    print (-, yj-1)
Set (i, j) := T(i, j)
until M(i, j) = 0.
```

### *C. Algoritma Needlemen-Wunsch*

1. Pengertian

Algoritma *Needlemen-Wunsch* mengimplementasikan pemrograman dinamis yang pertama kali untuk membandingkan urutan-urutan DNA, RNA, atau protein, dengan cara membagi urutan-urutan DNA yang besar menjadi lebih kecil dan mencari solusinya [3].

2. Pseudo Code

Pseoudo Code untuk membuat matriks dari *input string* A dan B adalah sebagai berikut [4] :

```
Procedure Fmatriks(input A, B : string, input/output F : matriks) {I.S : Matriks kosong dengan ukuran length(A)+1 x length(B)+1 F.S : Matriks terisi sesuai ketentuan algoritma NeedlemanWunsch}
Algoritma
for i=0 to length(A)
    F(i,0) <- 0
```

```
   Endfor
   for j=0 to length(B)
       F(0,j) <- 0
   Endfor
   for i=1 to length(A)
       for j = 1 to length(B)
       {
             Value1 <- F(i-1,j-1) + S(A(i),
       B(j))
             Value2 <- F(i-1, j) + d
             Value3 <- F(i, j-1) + d
             F(i,j) <- max(Value1, Value2,
       Value3)
       }
       Endfor
   Endfor
End-Algoritma
```

Pseudo Code untuk menyusun jalur dan mencocokannya adalah sebagai berikut [4] :

```
Procedure ScorePath(input A, B : string,
input F : matriks, output AlignmentA,
AlignmentB : string) {I.S : Matriks sudah
terisi F.S : Menciptakan jalur untuk
penjajaran, AlignmentA dan AlignmentB berisi
string dengan nilai kecocokan maksimum}
Algoritma
AlignmentA <- ""
AlignmentB <- ""
i <- length(A)
j <- length(B)
while (i > 0 and j > 0)
{
   Score <- F(i,j)
   ScoreDiag <- F(i - 1, j - 1)
   ScoreUp <- F(i - 1, j)
   ScoreLeft <- F(i, j - 1)
   if (Score == ScoreDiag + S(A(i), B(j)))
   {
       AlignmentA <- A(i) + AlignmentA
       AlignmentB <- B(j) + AlignmentB
       i <- i - 1
       j <- j - 1
       }
   Endif
   else if (Score == ScoreLeft + d)
   {
   AlignmentA <- "-" + AlignmentA
   AlignmentB <- B(j) + AlignmentB
   j <- j - 1
   }
   Endif
   Else if (Score == ScoreUp + d)
   {
       AlignmentA <- A(i) + AlignmentA
       AlignmentB <- "-" + AlignmentB
       i <- i - 1
   }
   Endif
}
Endwhile
while (i >= 0)
{
   AlignmentA <- A(i) + AlignmentA
   AlignmentB <- "-" + AlignmentB
i <- i - 1
}
Endwhile
while (j >= 0)
{
   AlignmentA <- "-" + AlignmentA
   AlignmentB <- B(j) + AlignmentB
   j <- j - 1
}
Endwhile
End-Algoritma
```

## III. ANALISIS SEQUENCE ALIGNMENT

*Sequence Alignment* memiliki fungsi untuk mensejajarkan dan menyimpulkan apakah dua DNA yang telah diurutkan memiliki hubungan yang erat atau tidak, baik secara fungsional, struktural ataupun *evelusioner*.

Disini kita akan mencoba untuk menyelaraskan dua urutan DNA dengan menggunakan Algoritma *Smith-Waterman* atau sering disebut dengan *Local Sequence Alignment* dan dengan menggunakan Algoritma *Needlemen-Wunsch* atau yang sering disebut dengan *Global Sequence Alignment*.

**Problem 1**

Misalkan kita punya dua urutan DNA sebagai berikut :
1. GTCAAGTA
2. TGCCATGA

**Local Sequence Alignment (Smith-Waterman Algorithm)**

Tahapan-tahapan yang harus dilakukan dalam melakukan *local sequence alignment* adalah sebagai berikut [1] :

1. Menginisialisasi skema *scoring* (*Match*, *Mismatch* dan *Gap*)
2. Membuat *Scoring Matrix*
3. *Scoring*
4. *Trace back*

Dimulai dengan menginisialisasi skema *scoring*, biasanya untuk skor *match* dan *mismatch* selalu kebalikannya. Misalnya untuk skor *match* adalah 2 maka untuk skor *mismatch* adalah -2, kita ambil contoh skema skor seperti dibawah ini :

1. *Match* = 3
2. *Mismatch* = -3
3. *Gap* = -2

Lalu langkah selanjutnya adalah membuat *Scoring Matrix*, salah satu contoh tampilan untuk *scoring matrix* adalah sebagai berikut :

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| T | | | | | | | | | |
| G | | | | | | | | | |
| C | | | | | | | | | |
| C | | | | | | | | | |
| A | | | | | | | | | |
| T | | | | | | | | | |
| G | | | | | | | | | |
| A | | | | | | | | | |

Setelah kita membuat *Scoring Matrix*, langkah selanjutnya adalah dengan mengisi *Scoring Matrix* tersebut berdasarkan skema skor yang telah kita buat sebelumnya. Dan nilai untuk kolom yang pertama dan baris yang pertama adalah 0.

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T | 0 | | | | | | | | |
| G | 0 | | | | | | | | |
| C | 0 | | | | | | | | |
| C | 0 | | | | | | | | |
| A | 0 | | | | | | | | |
| T | 0 | | | | | | | | |
| G | 0 | | | | | | | | |
| A | 0 | | | | | | | | |

Lalu cara mengisi sel-sel berikutnya yang kosong adalah sebagai berikut :

Untuk setiap sel, kita akan mencari nilai maksimum dari nilai linear horizontal, vertikal dan diagonal sel tersebut. Satu hal yang harus diketahui dari *local sequence alignment* ini adalah jika nilai maksimum sel tersebut adalah negatif maka hasilnya adalah 0.

Contoh cara Pengisian 2 baris pertama adalah sebagai berikut :

| | | G | T |
|---|---|---|---|
| | 0 | 0 | 0 |
| T | 0 | -3 **0** -2 | -2 |

Gambar diatas adalah cara *scoring* dari *local sequence alignment*, seperti yang telah dijelaskan diatas. *Scoring* yang akan kita lakukan adalah dengan cara mencari nilai maksimum dari diagonal (yang berwarna merah), linear horizontal (hijau) dan linear vertikal (hijau).

Nilai Maksimum = Max( skor diagonal + sel sebelumnya, skor horizontal + sel sebelumnya, skor vertikal + sel sebelumnya)

Nilai Maksimum = Max( -3 + 0, -2 + 0, -2 + 0)

Nilai Maksimum = Max(-3,-2,-2)

Nilai Maksimum = 0 (ingat, jika nilai maksimum adalah negatif, dalam *local sequence alignment* maka nilainya diubah menjadi 0)

Kenapa skor diagonal -3, karena kita liat skema nilainya jika *mismatch* maka nilainya -3, dan yang kita bandingkan adalah G dengan T artinya *mismatch* atau tidak sama. Dan kenapa untuk vertikal dan horizontal nilainya -2, karena untuk vertikal kita membandingkan G dengan *Gap* artinya nilainya adalah *Gap* yaitu -2, sama halnya dengan Horizontal kita membandingkan T dengan Gap maka nilainya adalah *Gap* yaitu -2.

Maka untuk menghitung sel baris berikutnya adalah sebagai berikut :

| | | G | T |
|---|---|---|---|
| | 0 | 0 | 0 |
| T | 0 | 3 **0** -2 | **3** -2 |

Nilai Maksimum = Max( skor diagonal + sel sebelumnya, skor horizontal + sel sebelumnya, skor vertikal + sel sebelumnya)

Nilai Maksimum = Max(3 + 0, -2 + 0, -2 + 0)

Nilai Maksimum = Max(3,-2,-2)

Nilai Maksimum = 3

dan seterusnya seperti itu, maka hasil yang didapatkan dari *scoring* adalah sebagai berikut :

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 3 | 1 |
| G | 0 | 3 | 1 | 0 | 0 | 0 | 3 | 1 | 0 |
| C | 0 | 1 | 0 | 4 | 2 | 0 | 1 | 0 | 0 |
| C | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 0 |
| A | 0 | 0 | 0 | 1 | 6 | 4 | 2 | 0 | 3 |
| T | 0 | 0 | 3 | 1 | 4 | 3 | 1 | 5 | 3 |
| G | 0 | 3 | 1 | 0 | 2 | 1 | 6 | 4 | 2 |
| A | 0 | 1 | 0 | 0 | 3 | 5 | 4 | 3 | 7 |

Tabel diatas adalah *scoring matrix* yang telah diisi semuanya, dan untuk langkah selanjutnya adalah *trace back*. *Trace back* sendiri untuk *local sequence alignment* dilakukan dengan memulainya dari nilai sel yang paling terbesar dan menyusurinya sampai dengan ditemukannya nilai 0. Maka hasil yang akan kita dapatkan adalah sebagai berikut :

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 3 | 1 |
| G | 0 | 3 | 1 | 0 | 0 | 0 | 3 | 1 | 0 |
| C | 0 | 1 | 0 | 4 | 2 | 0 | 1 | 0 | 0 |
| C | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 0 |
| A | 0 | 0 | 0 | 1 | 6 | 4 | 2 | 0 | 3 |
| T | 0 | 0 | 3 | 1 | 4 | 3 | 1 | 5 | 3 |
| G | 0 | 3 | 1 | 0 | 2 | 1 | 6 | 4 | 2 |
| A | 0 | 1 | 0 | 0 | 3 | 5 | 4 | 3 | 7 |

Dari tabel diatas maka hasil *Sequence Alignment*-nya adalah :

```
C   A   A   G   T   A
|   |   |   |   |   |
C   A   T   G   -   A
```

Setelah kita mendapatkan kesejajarannya, mari kita liat berapakah nilai maksimum dari kesejajaran tersebut dengan menggunakan skema skor yang telah kita tentukan sebelumnya

Nilai Kecocokan maksimum = 3 + 3 – 3 + 3 – 2 + 3

Nilai Kecocokan Maksimum = 7.

**Global Sequence Alignment (Needlemen-Wunsch Algorithm)**

setelah kita mencari *Alignment* dengan algoritma *Smith-Waterman*, sekarang kita coba mencari kesejajaran dua string diatas dengan algoritma *Needlemen-Wunsch*.

Tahapan-Tahapannya dan proses *scoring* sama dengan *local sequence alignment* yang membedakannya adalah jika ada nilai negatif maka tuliskan saja nilai negatif tersebut.

Skema skor yang akan kita gunakan adalah sebagai berikut :

1. *Match* = +1
2. *Mismatch* = -1
3. *Gap* = -1

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| T | | | | | | | | | |
| G | | | | | | | | | |
| C | | | | | | | | | |
| C | | | | | | | | | |
| A | | | | | | | | | |
| T | | | | | | | | | |
| G | | | | | | | | | |
| A | | | | | | | | | |

Untuk mengisi baris dan kolom yang pertama dimulai dengan 0 lalu untuk mengisi baris, kita tambahkan dengan *gap* karena tidak adanya sel top ataupun diagonal, begitu pula utuk kolom. maka hasilnya adalah :

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 |
| T | -1 | | | | | | | | |
| G | -2 | | | | | | | | |
| C | -3 | | | | | | | | |
| C | -4 | | | | | | | | |
| A | -5 | | | | | | | | |
| T | -6 | | | | | | | | |
| G | -7 | | | | | | | | |
| A | -8 | | | | | | | | |

Dan untuk mengisi kolom dan baris yang lainnya sama dengan cara yang dilakukan dalam *local sequence alignment*, yaitu mencari nilai terbesar dari diagonal, vertikal dan horizontal. Maka kita akan mendapatkan hasilnya sebagai berikut :

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 |
| T | -1 | -1 | 0 | -1 | -2 | -3 | -4 | -5 | -6 |
| G | -2 | 0 | -1 | -1 | -2 | -3 | -2 | -3 | -4 |
| C | -3 | -1 | -1 | 0 | -1 | -2 | -3 | -3 | -4 |
| C | -4 | -2 | -2 | 0 | -1 | -2 | -3 | -4 | -4 |
| A | -5 | -3 | -3 | -1 | 1 | 0 | -1 | -2 | -3 |
| T | -6 | -4 | -2 | -2 | 0 | 0 | -1 | 0 | -1 |
| G | -7 | -5 | -3 | -3 | -1 | -1 | 1 | 0 | -1 |
| A | -8 | -6 | -4 | -4 | -2 | 0 | 0 | 0 | 1 |

*Trace back* yang dilakukan dalam *global sequence alignment* bukanlah dari nilai sel yang terbesar, melaikan dari ujung bawah, seperti pada gambar berikut ini :

| | | G | T | C | A | A | G | T | A |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 |
| T | -1 | -1 | 0 | -1 | -2 | -3 | -4 | -5 | -6 |
| G | -2 | 0 | -1 | -1 | -2 | -3 | -2 | -3 | -4 |
| C | -3 | -1 | -1 | 0 | -1 | -2 | -3 | -3 | -4 |
| C | -4 | -2 | -2 | 0 | -1 | -2 | -3 | -4 | -4 |
| A | -5 | -3 | -3 | -1 | 1 | 0 | -1 | -2 | -3 |
| T | -6 | -4 | -2 | -2 | 0 | 0 | -1 | 0 | -1 |
| G | -7 | -5 | -3 | -3 | -1 | -1 | 1 | 0 | -1 |
| A | -8 | -6 | -4 | -4 | -2 | 0 | 0 | 0 | 1 |

Kesejajaran yang dibentuk adalah sebagai berikut :

```
G  T  C  A  A  G  T  A
|  |  |  |  |  |  |  |
G  C  C  A  T  G  -  A
```

Nilai Kecocokan Maksimum = 1 – 1 + 1 + 1 – 1 + 1 – 1 + 1

Nilai Kecocokan Maksimum = 2

**Problem 2**

Misalkan kita memiliki 2 sequence DNA berikut ini :

1. Sequence 1 : GGCTTAATCC
2. Sequence 2 : AATCCGGAAT

Dengan skema skor seperti dibawah ini :

1. *Match :* +2
2. *Mismatch :* -2
3. *Gap :* -1

**Global Sequence Alignment**

Maka *Scoring Matrix* yang akan kita miliki adalah sebagai berikut :

| | | G | G | C | T | T | A | A | T | C | C |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -10 |
| A | -1 | -2 | -3 | -4 | -5 | -6 | -3 | -4 | -5 | -6 | -7 |
| A | -2 | -3 | -4 | -5 | -6 | -7 | -4 | -1 | -2 | -3 | -4 |
| T | -3 | -4 | -5 | -6 | -3 | -4 | -5 | -2 | 1 | 0 | -1 |
| C | -4 | -5 | -6 | -3 | -4 | -5 | -6 | -3 | 0 | 3 | 2 |
| C | -5 | -6 | -7 | -2 | -3 | -4 | -5 | -4 | -1 | 2 | 5 |
| G | -6 | -3 | -4 | -3 | -4 | -5 | -6 | -5 | -3 | 1 | 4 |
| G | -7 | -2 | -1 | -2 | -3 | -4 | -5 | -6 | -3 | 0 | 3 |
| A | -8 | -3 | -2 | -3 | -4 | -5 | -2 | -3 | -4 | -1 | 2 |
| A | -9 | -4 | -3 | -4 | -5 | -6 | -3 | 0 | -1 | -2 | 1 |
| T | -10 | -5 | -4 | -5 | -2 | -3 | -4 | -1 | 2 | 1 | 0 |

Maka kesejajaran yang dibentuk adalah sebagai berikut :

```
A  A  T  C  C  -  -  -  -  -
|  |  |  |  |  |  |  |  |  |
A  A  T  C  C  G  G  A  A  T
```

Dengan nilai kecocokan maksimumnya adalah

2 + 2 + 2 + 2 + 2 -1 -1 -1 -1 -1 = 5

**Local Sequence Alignment**

Maka *Scoring Matrix* yang akan kita dapatkan adalah sebagai berikut :

| | | G | G | C | T | T | A | A | T | C | C |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 1 | 0 | 0 |
| A | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 4 | 3 | 0 | 0 |
| T | 0 | 0 | 0 | 0 | 2 | 2 | 1 | 3 | 6 | 5 | 4 |
| C | 0 | 0 | 0 | 2 | 1 | 1 | 0 | 2 | 5 | 8 | 7 |
| C | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 1 | 4 | 7 | 10 |
| G | 0 | 2 | 2 | 1 | 0 | 0 | 0 | 0 | 3 | 6 | 9 |
| G | 0 | 2 | 4 | 3 | 2 | 1 | 0 | 0 | 2 | 5 | 8 |
| A | 0 | 1 | 3 | 2 | 1 | 0 | 3 | 2 | 1 | 4 | 7 |
| A | 0 | 0 | 2 | 1 | 0 | 0 | 2 | 5 | 4 | 3 | 6 |
| T | 0 | 0 | 1 | 0 | 0 | 2 | 1 | 4 | 7 | 6 | 5 |

Kesejajaran yang dibentuk dari s*coring matrix* diatas adalah sebagai berikut :

```
A  A  T  C  C
|  |  |  |  |
A  A  T  C  C
```

Dengan nilai kecocokan maksimumnya adalah

2 + 2 + 2 + 2 + 2 = 10

Dari problem 2 ini kita dapat melihat perbedaan yang sangat jelas antara *local sequence alignment* dan *global sequence alignment*. Dimana dengan 2 urutan yang berbeda satu sama lain tidak cocok untuk menggunakan *global sequence alignment* karena akan menghasilkan banyak gap. Gap sendiri menandakan adanya ketidakmiripan antara dua sequence tersebut, dan juga menandakan bahwa sequence yang memiliki gap telah terjadi penambahan atau penghapusan DNA. Untuk kasus ini metode yang cocok adalah dengan menggunakan *local sequence alignment* karena untuk *local* hanya mencari *sub-sequence* yang sama diantara kedua urutan tersebut, sedangkan untuk *global* dia membandingkan semua urutan.

**Perbedaan Algoritma Smith-Waterman dan Algoritma Needlemen-Wunsch**

Perbedaan local sequence alignment dengan global sequence alignment dari sisi penerapan algoritmanya adalah sebagai berikut [2] :

| | Smith-Waterman | Needlemen-Wunsch |
|---|---|---|
| Inisialisasi | Baris dan kolom pertama diisi dengan 0 | Baris dan kolom pertama nilainya tergantung dari nilai *Gap* |
| *Scoring* | Jika ada nilai negatif maka ubah menjadi 0 | *Score* bisa negatif |
| *Trace back* | Dimulai dari nilai sel yang paling tinggi dan berakhir sampe ditemukannya 0 | Dimulai dari sel ujung kanan bawah sampai dengan ditemukannya sel *origin*-nya |

Perbedaan yang paling mendasar dari kedua algoritma diatas adalah untuk *Local Sequence Alignment* menemukan segmen-segmen yang mirip dari masing masing urutan, sedangkan untuk *Global Sequence Alignment* membandingkan keseluruhan urutan-urutan tersebut.

Manfaat dari adanya algoritma ini adalah untuk *local sequence alignment* sendiri digunakan untuk mencari domain homolog dari gen non-homolog atau mencari domain dari dua protein . sedangkan untuk *global sequence alignment* sering digunakan untuk membandingkan gen homolog, atau membandingkan 2 gen dengan fungsi yang sama di dalam manusia dan tikus.

## IV. KESIMPULAN

*Sequence Alignment* dalam bioinformatik adalah cara mengurutkan urutan-urutan DNA, RNA dan Protein untuk mengidentifikasi kesamaannya. Ada dua cara untuk melakukan *Alignment* yaitu dengan menggunakan algoritma *Smith-Waterman* atau yang biasa disebut dengan *Local Sequence Alignment* dan algoritma *Needlemen-Wunsch* atau sering disebut juga dengan *Global Sequene Alignment*.

Fungsi dari dua algoritma tersebut sama yaitu untuk melakukan *Alignment* dari urutan-urutan DNA, RNA atau protein tapi memiliki tujuan yang berbeda. untuk *local sequence alignment* sendiri digunakan untuk mencari domain homolog dari gen non-homolog atau mencari domain dari dua protein . sedangkan untuk *global sequence alignment* sering digunakan untuk membandingkan gen homolog, atau membandingkan 2 gen dengan fungsi yang sama di dalam manusia dan tikus. Homolog sendiri adalah organ-organ berbagai mahluk hidup yang mempunyai bentuk yang sama, struktur yang sama tetapi memiliki fungsi yang berbeda misalnya seperti tangan manusia dan tangan kucing. Sedangkan untuk non-homolog sendiri, jika ada organ yang bentuk ataupun ukurannya tidak sama maka disebut non-homolog.

## DAFTAR PUSTAKA


[1] Smith-Waterman Algorithm - Local Alignment of Sequences, http://vlab.amrita.edu/?sub=3&brch=274&sim=1433&cnt=1

[2] Smith, Temple F. & Waterman, Michael S. (1981). "Identification of Common Molecular Subsequences" . *Journal of Molecular Biology*. **147**: 195–197

[3] Needleman, Saul B. & Wunsch, Christian D. (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". *Journal of Molecular Biology*. **48** (3): 443–53.

[4] M.R. Firdaus, S.W. Putri M.F. Rasyid. 2006. Penerapan Algoritma Needleman-Wunsch sebagai Salah Satu Implementasi Program Dinamis pada Pensejajaran DNA dan Protein. Makalah STMIK, bandung

[5] Bambang, Alif. 2017. Penerapan Algoritma Program Dinamis pada Penyejajaran Sekuens dengan Algoritma Smith–Waterman. Makalah IF2211 Strategi Algoritma, Bandung.