Penerapan Algoritma Needleman-Wunsch sebagai Salah Satu Implementasi Program Dinamis pada Pensejajaran DNA dan Protein

Muhamad Reza Firdaus Zen[1], Sila Wiyanti Putri[2], Muhamad Fajrin Rasyid[3]

Laboratorium Ilmu dan Rekayasa Komputasi
Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail : if14048@students.if.itb.ac.id[1], if14051@students.if.itb.ac.id[2], if14055@students.if.itb.ac.id[3]

Abstrak

Pada saat ini, pengujian DNA sudah menjadi salah satu metode paling ampuh dalam membuktikan apakah seseorang berkerabat langsung (keturunan dari seseorang lain) atau menentukan apakah suatu organisme merupakan anggota dari spesies tertentu. Dalam proses pengujian kekerabatan langsung, DNA dari kedua orang tersebut disejajarkan. Apabila ditemukan ketidakcocokan melebihi batas toleransi tertentu, maka dapat dipastikan bahwa kedua orang tersebut tidak mungkin memiliki hubungan kekerabatan langsung (orang tua dan anak). Sedangkan untuk pengujian keanggotaan spesies, DNA organisme uji disejajarkan dengan rangkaian umum DNA spesies dugaan. Salah satu metode yang digunakan dalam pensejajaran DNA adalah algoritma *Needleman-Wunsch*, yang merupakan perluasan algoritma pencocokan string pada teks atau barisan sebagai salah satu implementasi program dinamis. Algoritma *Needleman-Wunsch* dapat diterapkan karena pada dasarnya DNA adalah rangkaian lanjar ganda gugus-gugus karbon, dimana setiap gugus karbon dapat dianalogikan sebagai karakter, sehingga DNA dapat dianalogikan sebagai rangkaian karakter atau string. Pendekatan serupa juga dapat diterapkan untuk mensejajarkan jenis-jenis protein, yang merupakan rangkaian asam amino, dimana asam amino, seperti DNA, merupakan rangkaian gugus-gugus karbon. Pada makalah ini akan dibahas bagaimana cara kerja algoritma *Needleman-Wunsch*, penerapannya dalam mensejajarkan DNA dan protein, serta bagaimana kompleksitas ruang dan waktu algoritma tersebut.

***Kata kunci**: algoritma Needleman-Wunsch, program dinamis, DNA, protein.*

1. Pendahuluan

DNA (*Deoxyribo Nucleic Acid*) merupakan struktur informasi unik terkecil yang dimiliki oleh setiap organisme, yang diwariskan dari generasi ke generasi. DNA berbentuk rangkaian lanjar ganda yang tersusun atas empat jenis gugus karbon, yaitu Adenin (A), Sitosin (C), Timin (T), dan Guanin (G). Hal ini berarti, DNA dapat dinyatakan sebagai rangkaian karakter (string) dari empat kemungkinan karakter A, C, G, dan T.

Pada sebagian besar organisme, pewarisan tersebut tidak dapat berlangsung 100% sempurna, karena terdapat faktor pembatas lingkungan, seperti mutasi dan degenerasi gen. Meskipun demikian, selalu terjamin bahwa sejumlah rangkaian DNA dalam batas tertentu, bergantung pada jenis organisme, diwariskan oleh suatu organisme kepada keturunan-keturunannya.

Sejumlah organisme dengan tingkat keserupaan DNA tertentu membentuk suatu spesies. Dengan kata lain, setiap organisme suatu spesies pasti memiliki kecocokan DNA dalam batas tertentu dengan rangkaian umum DNA spesies tersebut. Oleh karena itu, ilmuwan dapat mengkategorikan suatu organisme ke dalam suatu spesies baru, apabila DNA organisme tersebut tidak cocok dalam setiap batas tertentu dengan DNA spesies-spesies yang telah terdefinisi sebelumnya.

Sementara itu, protein merupakan rangkaian lanjar asam amino. Asam amino, sama halnya dengan DNA, merupakan rangkaian tiga gugus karbon, tiap gugus karbon merupakan salah satu dari gugus Adenin (A), Sitosin (C), Urasil (U), atau Guanin (G).

2. Pensejajaran DNA dan Protein

Dalam proses pencocokkan dua atau lebih DNA, tidak mungkin ditemukan kesamaan 100% karena seperti dijelaskan sebelumnya, DNA bersifat unik. Namun, karena panjang DNA (jumlah rangkaian gugus karbon) dua atau lebih organisme tidak selalu sama, proses yang digunakan bukanlah ‘pembandingan’, melainkan ‘pensejajaran’. Hal

yang sama juga diterapkan dalam proses pencocokkan protein.

Pensejajaran dua buah DNA adalah memberikan nilai kecocokan kedua DNA tersebut. Semakin tinggi nilainya, berarti kedua DNA tersebut semakin mirip satu sama lain, dan sebaliknya, semakin rendah nilainya, berarti kedua DNA tersebut semakin tidak mirip. Apabila panjang salah satu DNA (sebut DNA A) lebih panjang daripada DNA lainnya (sebut DNA B), maka kita dapat menyisipkan celah-celah kosong pada DNA B sehingga panjang DNA B menjadi sama dengan DNA A (celah-celah kosong tersebut disimbolkan dengan "-").

Contoh-contoh pensejajaran 2 buah DNA :

1) DNA A = AGCAAGTGGAC, dan DNA B = ACCAAGTGGAC, maka kedua DNA tersebut hanya berbeda pada karakter kedua. Dengan demikian, dapat disimpulkan bahwa DNA A dan DNA B mirip, sehingga nilai kecocokan DNA A dan B tinggi.
2) DNA A = AGCAAGTGGAC, dan DNA B = GCTGTCCACGT, maka kedua DNA tersebut berbeda pada semua karakter. Dengan demikian, dapat disimpulkan bahwa DNA A dan DNA B sangat tidak mirip, sehingga nilai kecocokan DNA A dan B rendah.
3) DNA A = AGCAAGTGGAC, dan DNA B = ACAAGTGGAC, DNA A lebih panjang dari DNA B, sehingga kita harus menyisipkan karakter celah kosong ("-") pada DNA B. Jika kita menyisipkan karakter celah kosong ("-") sebelum karakter kedua DNA B, maka DNA B = A-CAAGTGGAC. Dengan penyisipan ini, DNA A dan DNA B hanya berbeda pada karakter kedua. Karena kita dapat menyisipkan celah-celah sedemikian hingga DNA A dan B mirip, dapat disimpulkan bahwa nilai kecocokan DNA A dan B tinggi.

3. Algoritma Needleman-Wunsch

3.1. Pengertian

Algoritma Needleman-Wunsch merupakan perluasan algoritma pencocokan string pada teks atau barisan sebagai salah satu implementasi program dinamis. Algoritma ini merupakan salah satu algoritma yang digunakan untuk menentukan tingkat kesamaan atau kecocokan dua buah teks.

3.2. Fungsi dan Nilai yang Digunakan

Prosedur algoritma Needleman-Wunsch adalah sebagai berikut. Pertama-tama, tetapkan nilai untuk setiap kecocokan karakter, ketidakcocokan karakter, serta nilai penalti apabila salah satu karakter dari kedua teks yang dibandingkan digeser sehingga diganti dengan karakter celah kosong. Nilai kecocokan karakter boleh sama untuk setiap karakter, boleh tidak (bergantung definisi yang diinginkan). Nilai kecocokan (apabila dua karakter sama) haruslah ditetapkan sebagai suatu nilai positif. Hal ini disebabkan dua teks dikatakan semakin mirip jika nilai kecocokannya tinggi, sementara nilai kecocokan kedua teks tinggi apabila nilai kecocokan tiap karakter tinggi.

Sebaliknya, nilai ketidakcocokan ditetapkan sebagai nilai negatif atau nol. Nilai ketidakcocokan karakter juga harus berlaku simetris, artinya jika ketidakcocokan karakter a dan b bernilai -10, maka ketidakcocokan karakter b dan a juga harus bernilai -10. Serupa dengan nilai kecocokan, nilai ketidakcocokan harus bernilai negatif atau nol karena ketidakcocokan karakter mengurangi kemiripan kedua teks. Adapun nilai penalti apabila salah satu karakter dari kedua teks yang dibandingkan digeser sehingga diganti dengan karakter celah kosong juga harus bernilai negatif. Hal ini disebabkan kita memerlukan upaya tambahan untuk menggeser karakter-karakter setelah celah yang disisipkan.

Setelah ketiga nilai tadi ditetapkan, langkah selanjutnya adalah membentuk matriks berukuran jumlah baris = panjang teks pertama + 1, dan jumlah kolom = panjang teks kedua + 1.Tambahan satu kolom dan satu baris diperlukan untuk menjadi basis bagi kolom-kolom dan baris-baris selanjutnya. Apabila penomoran baris dan kolom dimulai dari nol, maka elemen matriks pada baris ke-i dan kolom ke-j (disimbolkan dengan F_{ij}) menyatakan nilai kecocokan maksimum hingga karakter ke-i teks pertama dan karakter ke-j teks kedua.

Basis untuk algoritma Needleman-Wunsch adalah :
$F_{i0} = 0$ untuk i = 1 sampai dengan length(A)
$F_{0j} = 0$ untuk j = 1 sampai dengan length(B)

Rekurens untuk algoritma Needleman-Wunsch adalah mengisi semua elemen pada baris ke-1 sampai dengan baris ke-length(A), dengan aturan :
$F_{ij} = \max(F_{i-1,j-1} + S(A_iB_j),\ F_{i,j-1} + d\ ,\ F_{i-1,j} + d\ )$,
dengan $S(A_iB_j)$ adalah nilai perbandingan (kecocokan atau ketidakcocokan) antara karakter ke-i teks A dan karakter ke-j teks B, dan d adalah nilai penalti karena salah satu karakter di antara karakter ke-i teks A dan karakter ke-j teks B diganti oleh celah.

Dalam setiap pengisian elemen matriks, Fungsi F_{ij} selalu mendasarkan pada solusi optimum elemen-elemen pada tahap sebelumnya (yaitu pada baris dan kolom sebelumnya). Dengan demikian, setiap elemen matriks F_{ij} akan berisi nilai kecocokan maksimum hingga karakter ke-i teks A dan karakter ke-j teks B. Di akhir pengisian nilai elemen-elemen matriks, nilai elemen matriks pada baris ke-

length(A) dan kolom ke-length(B) akan menyatakan nilai kecocokan maksimum antara teks A dan teks B.

Setelah matriks *F* terbentuk, maka langkah selanjutnya adalah menentukan jalur dari elemen pada baris ke-length(A) dan kolom ke-length(B) (misal F_{ij}) hingga elemen pada baris ke-0 dan kolom ke-0 (misal F_{00}). Hal ini dapat dianalogikan dengan pencarian jalur dari simpul solusi ke simpul awal (akar) pada pohon yang dibangkitkan dengan algoritma *Breadth First Search* (BFS). Pencarian jalur ini dimungkinkan karena apabila kita mengambil sembarang elemen matriks *F* (misal F_{ij}), maka kita dapat membandingkannya dengan nilai $F_{i-1,j-1} + S(A_iB_j)$, $F_{i,j-1} + d$, dan $F_{i-1,j} + d$ untuk menentukan dari mana nilai F_{ij} berasal. Apabila dalam mencari jalur dari elemen F_{ij}, terdapat lebih dari satu asal nilai F_{ij}, maka hal ini mengakibatkan kita dapat menemukan lebih dari satu jalur menuju elemen F_{00}. Pada kasus demikian, terdapat lebih dari satu solusi penyelesaian.

Langkah terakhir setelah jalur dari elemen pada baris ke-length(A) dan kolom ke-length(B) (misal F_{ij}) hingga elemen pada baris ke-0 dan kolom ke-0 (misal F_{00}) terbentuk adalah mencocokkan karakter-karakter kedua teks yang bersesuaian. Prosedur ini, dan juga prosedur-prosedur sebelumnya, akan lebih jelas ditunjukkan pada Pseudo-code pada bagian 3.3 dan penerapan pada bagian 4.

3.3. Pseudo-code

Pseudo-code untuk membuat matriks *F* berdasarkan input teks A dan B adalah sebagai berikut :

```
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 Needleman-
Wunsch, yaitu mengisi tiap elemen
matriks dengan basis dan rekurens
seperti dijelaskan pada bagian 3.2.
}
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 dari elemen pada baris ke-length(A) dan kolom ke-length(B) (misal F_{ij}) hingga elemen pada baris ke-0 dan kolom ke-0 (misal F_{00}), sekaligus mencocokkan karakter-karakter A dan B adalah sebagai berikut :

```
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
```

Keterangan :
Dalam Pseudo-code ini, hanya diambil satu jalur solusi saja, dengan prioritas membandingkan kesamaan setiap elemen matriks dengan elemen di sebelah kiri atasnya, kemudian elemen di sebelah kirinya, dan terakhir elemen di sebelah atasnya.

3.4. Kompleksitas

Kompleksitas waktu dan ruang algoritma Needleman-Wunsch adalah sebanding dengan besar matriks yang dibutuhkan untuk menampung elemen-elemen berisi nilai kecocokan kedua teks yang dibandingkan. Apabila panjang teks pertama adalah *m* dan panjang teks kedua adalah *n*, maka ukuran matriks yang dibutuhkan adalah *m* x *n*. Dengan demikian, kompleksitasnya adalah *O*(*mn*). Adapun algoritma Brute Force untuk masalah ini (dengan asumsi bahwa *m* > n) memiliki kompleksitas *O*(*m!/n!*). Artinya, untuk nilai *m* dan *n* semakin besar, algoritma Needleman-Wunsch lebih mangkus daripada algoritma Brute Force.

4. Penerapan Algoritma Needleman-Wunsch dalam Pensejajaran DNA dan Protein

Tinjau DNA A = ATC dan DNA B = AGCT. Akan dihitung berapa nilai kecocokan maksimal kedua DNA tersebut, dan susunan dengan celah seperti apa yang dibutuhkan untuk mencapai nilai kecocokan maksimum tersebut. Definisikan nilai kecocokan setiap karakter adalah 5, nilai ketidakcocokan setiap karakter adalah -2, dan nilai penalti karena suatu karakter diganti celah adalah -3. Misalkan ditetapkan pula batas minimal nilai kecocokan kedua DNA tersebut agar termasuk dalam spesies yang sama adalah 4. Dengan demikian, apabila nilai kecocokan maksimal kedua DNA tersebut lebih besar atau sama dengan 4, maka dapati disimpulkan bahwa A dan B merupakan satu spesies.

Bentuk suatu matriks berukuran jumlah kolom = panjang DNA A + 1, dan jumlah baris = panjang DNA B + 1. Jadi, matriks yang terbentuk memiliki jumlah kolom = 4 + 1 = 5. Isi setiap elemen pada baris ke-0 dan kolom ke-0 dengan 0.

	0	A	G	C	T
0	0	0	0	0	0
A	0				
T	0				
C	0				

Selanjutnya, isi elemen pada baris ke-1 dan kolom ke-1 sesuai aturan Needleman-Wunsch. $F_{00} = F_{10} = F_{01} = 0$. Sementara itu, karena karakter ke-1 DNA A = karakter ke-1 DNA B, maka $S(A_1B_1) = 5$. Oleh karena itu, $F_{11} = \max(F_{00} + S(A_1B_1), F_{10} + d , F_{01} + d) = \max(0 + 5, 0 + (-3) , 0 + (-3)) = \max(5, -3 ,-3) = 5$.

	0	A	G	C	T
0	0	0	0	0	0
A	0	5			
T	0				
C	0				

Selanjutnya, isi elemen pada baris ke-1 dan kolom ke-2. $F_{01} = F_{02} = 0$, sementara $F_{11} = 5$. Sementara itu, karena karakter ke-1 DNA A tidak sama dengan karakter ke-2 DNA B, maka $S(A_1B_1) = -2$. Oleh karena itu, $F_{11} = \max(F_{01} + S(A_1B_2), F_{11} + d , F_{02} + d) = \max(0 + (-2), 5 + (-3) , 0 + (-3)) = \max(-2, 2 ,-3) = 2$.

	0	A	G	C	T
0	0	0	0	0	0
A	0	5 ←	2		
T	0				
C	0				

Apabila proses ini diteruskan, maka kita dapat memperoleh matriks yang merepresentasikan nilai kecocokan DNA A dan B sebagai berikut :

	0	A	G	C	T
0	0	0	0	0	0
A	0	5 ←	2 ←	-1	-2
T	0	2	3 ←	0	4
C	0	-1	0	8 ←	5

Karena elemen pada baris ke-length(A) dan kolom ke-length(B) bernilai 5, maka nilai kecocokan maksimum kedua DNA tersebut adalah 5. Dalam menentukan nilai setiap elemen matriks, dapat pula diberi tanda panah untuk mengetahui dari elemen mana nilai elemen matriks tersebut berasal. Hal ini

akan memudahkan dalam menentukan jalur dari elemen pada baris ke-length(A) dan kolom ke-length(B) (misal F_{ij}) hingga elemen pada baris ke-0 dan kolom ke-0 (misal F_{00}), karena kita cukup menelusuri tanda panah tersebut. Jalur untuk matriks tersebut tampak pada matriks di bawah ini :

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

Langkah terakhir adalah mencocokkan karakter-karakter DNA A dan B sesuai jalur pada matriks tersebut. Pertama, tinjau elemen pada baris ke-1 dan kolom ke-1. Karena baris ke-1 dan kolom ke-1 belum digunakan oleh kedua teks, maka karakter pertama DNA A setelah pensejajaran adalah 'A' dan karakter pertama DNA B setelah pensejajaran adalah 'A'. Selanjutnya, dari elemen tersebut kita berpindah ke elemen pada baris ke-2 kolom ke-2. Karena baris ke-2 dan kolom ke-2 belum digunakan oleh kedua teks, maka karakter kedua DNA A setelah pensejajaran adalah 'T' dan karakter kedua DNA B setelah pensejajaran adalah 'G'. Selanjutnya, dari elemen tersebut kita berpindah ke elemen pada baris ke-3 kolom ke-3. Karena baris ke-3 dan kolom ke-3 belum digunakan oleh kedua teks, maka karakter ketiga DNA A setelah pensejajaran adalah 'C' dan karakter ketiga DNA B setelah pensejajaran adalah 'C'. Selanjutnya, kita berpindah ke elemen pada baris ke-3 kolom ke-4. Baris ke-3 sudah digunakan oleh DNA A, sementara kolom ke-4 belum digunakan oleh DNA B. Oleh karena itu, karakter keempat DNA A setelah pensejajaran adalah '-' dan karakter keempat DNA B setelah pensejajaran adalah 'T'.

Dengan demikian, nilai kecocokan maksimum DNA A dan DNA B adalah 5, dan ini dicapai oleh DNA A = ATC- dan DNA B = AGCT. Karena nilai kecocokan maksimum kedua DNA tersebut lebih besar daripada batas minimal nilai kecocokan kedua DNA tersebut agar termasuk dalam spesies yang sama, maka dapat disimpulkan bahwa A dan B merupakan satu spesies.

5. Kesimpulan

Algoritma Needleman-Wunsch yang merupakan perluasan algoritma pencocokan string pada teks atau barisan sebagai salah satu implementasi program dinamis memiliki kompleksitas waktu dan ruang *O(mn)* dan lebih baik daripada algoritma Brute Force. Oleh karena itu, algoritma ini merupakan algoritma yang cukup baik apabila diterapkan dalam proses pensejajaran DNA dan protein. Di masa yang akan datang, pengembangan-pengembangan algoritma ini masih mungkin dilakukan, sebagai contoh meminimalisasi matriks yang diperlukan. Dengan demikian, proses pensejajaran DNA dan protein yang sangat diperlukan dalam menentukan kekerabatan, menentukan jenis spesies makhluk hidup, serta mengolah protein, dapat dijalankan dengan lebih baik dan mangkus.

6. Referensi

[1] L. Gonick dan M. Wheelis, *Kartun Biologi Genetika*, KPG, Gramedia, Jakarta, 2003

[2] Rinaldi Munir, *Diktat Kuliah IF2251 Strategi Algoritmik*, Program Studi Teknik Informatika ITB, 2005

[3] Pollar, Bergman, Stoye, Celniker, dan Eisen, Benchmarking Tools for the Alignment of Functional Noncoding DNA, http://rana.lbl.gov/AlignmentBenchmarking/methods.html, diakses tanggal 18 Mei 2006 pukul 17.30.

[4] Wikipedia, *Needleman-Wunsch Algorithm*, http://en.wikipedia.org/wiki/Needleman-Wunsch_algorithm.htm, diakses tanggal 17 Mei 2006 pukul 10.30

[5] Yun Xing, *Sequence Alignment Algorithm*, http://icl.pku.edu.cn/yujs/papers/pdf/SeqAli.pdf, diakses tanggal 18 Mei 2006 pukul 10.00

[6] Zhenghong Zhao, *Sequence Alignment*, http://www2.cs.uh.edu/~zhenzhao/Review/alignment.htm, diakses tanggal 18 Mei 2006 pukul 10.00