T – 1

Pensejajaran Rantai DNA Menggunakan Algoritma Dijkstra

Abduh Riski[1]

[1]Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Jember
riski.fmipa@unej.ac.id

Abstrak—*Deoxyribonucleic Acid* atau DNA adalah asam nukleat yang berisi informasi genetik dari setiap makhluk hidup. Pensejajaran rantai DNA merupakan metode untuk mengetahui tingkat kemiripan dua rantai DNA dari makhluk hidup yang berbeda. Dengan melakukan pensejajaraan rantai DNA dapat diperoleh informasi tentang fungsi, struktur atau evolusi kedua rantai DNA yang dibandingkan. Pada proses pensejajaran rantai DNA akan diperoleh skor-skor relasi dari kedua rantai DNA. Tujuan dari metode pensejajaran rantai DNA adalah mencari rangkaian rantai DNA dengan jumlah skor yang minimal. Pada penelitian ini proses pensejajaran rantai DNA akan dilakukan menggunakan algoritma djikstra. Algoritma djikstra adalah salah satu algoritma untuk mencari lintasan terpendek pada graf berbobot. Graf berbobot yang digunakan dibangkitkan berdasarkan skor-skor relasi rantai DNA yang dibandingkan, sehingga akan diperoleh graf dengan bobot positif. Hasil penelitian menunjukkan bahwa algoritma djikstra berhasil menghasilkan lintasan terpendek yang merupakan interpretasi rangkaian rantai DNA dengan skor minimal.

Kata kunci: Algoritma Dijkstra, DNA, Graf Berbobot, Pensejajaran

I. PENDAHULUAN

Pensejajaran rantai DNA merupakan teknik untuk mengetahui kemiripan dari himpunan rantai DNA [3]. Salah satu contoh penerapan metode ini adalah untuk identifikasi korban kecelakaan pesawat terbang. Prinsip dasar dari metode pensejajaran adalah dengan membandingkan satu persatu asam amino dari sebuah rantai DNA dengan asam amino dari rantai DNA yang lain. Sehingga diperoleh sebuah himpunan perluasan rantai DNA yang berisi perluasan rantai DNA dengan skor kemiripan yang besar atau dengan skor ketidak miripan yang kecil. Proses pensejajaran untuk rantai DNA berukuran panjang membutuhkan waktu yang relatif lama, sehingga dibutuhkan metode atau algoritma agar proses pensejaran yang dilakukan menjadi cepat. Beberapa metode yang pernah digunakan untuk melakukan pensejajaran di anataranya adalah pemrograman dinamis [2] dan algoritma genetika [3].

Dalam jurnal ini akan dilakukan pensejajaran rantai DNA menggunakan algoritma dijkstra. Algoritma dijkstra adalah algoritma yang sering dikenal sebagai algortima pencarian lintasan terpendek. Dengan terlebih dahulu membangun sebuah graf berbobot dari relasi antar rantai DNA dan kemudian menerapkan algortima dijkstra pada graf yang terbentuk, sehingga diharapkan dapat diperoleh lintasan terpendek yang merepresentasikan kesejajaran dari himpunan perluasan rantai DNA.

II. METODE PENELITIAN

A. Algoritma Dijkstra

Algoritma Dijkstra ditemukan oleh E. W. Dijkstra pada tahun 1959 [1], algoritma ini merupakan algortima pencarian graf untuk menyelesaikan permasalahan *single-source shortest path* pada sebuah graf berbobot tak-negatif. *Single-source shortest path* adalah permasalahan untuk mencari lintasan terpendek dari sebuah titik v ke semua titik lainnya dalam sebuah graf. Tetapi algoritma ini juga dapat digunakan untuk mencari lintasan terpendek dari sebuah titik ke sebuah titik lainnya dengan menghentikan proses algoritma ketika lintasan terpendek ke titik tujuan telah ditemukan.

Algoritma Dijkstra
Input: • Sebuah Graf berarah atau tak-berarah $G = (V, E)$ berbobot dan tidak memiliki *loops*. Order dari G adalah $n > 0$. Dimana titik-titiknya diberi label 1 hingga n, misalnya $V = \{1,2,3, \dots, n\}$. • Sebuah titik awal $s \in V$.

Output:

- List D yang berisi jarak, dimana $D[v]$ adalah jarak lintasan terpendek dari s ke v.
- List P yang berisi titik induk, dimana $P[v]$ adalah induk dari v, misalnya v bertetangga dengan $P[v]$.

1. $D \leftarrow [\infty, \infty, \ldots, \infty]$;
2. $D[s] \leftarrow 0$;
3. $P \leftarrow []$;
4. $Q \leftarrow V$; (list titik yang akan dikunjungi)
5. $while\ length(Q) > 0\ do$
6. Cari $v \in Q$ dimana $D[v]$ minimal;
7. $Q \leftarrow remove(Q, v)$; (hapus v dari Q)
8. $for\ each\ u \in adj(v) \cap Q\ do$
9. Jika $D[u] > D[v] + w(vu)$ then
10. $D[u] \leftarrow D[v] + w(vu)$;
11. $P[u] \leftarrow v$;
12. $return\ (D, P)$;

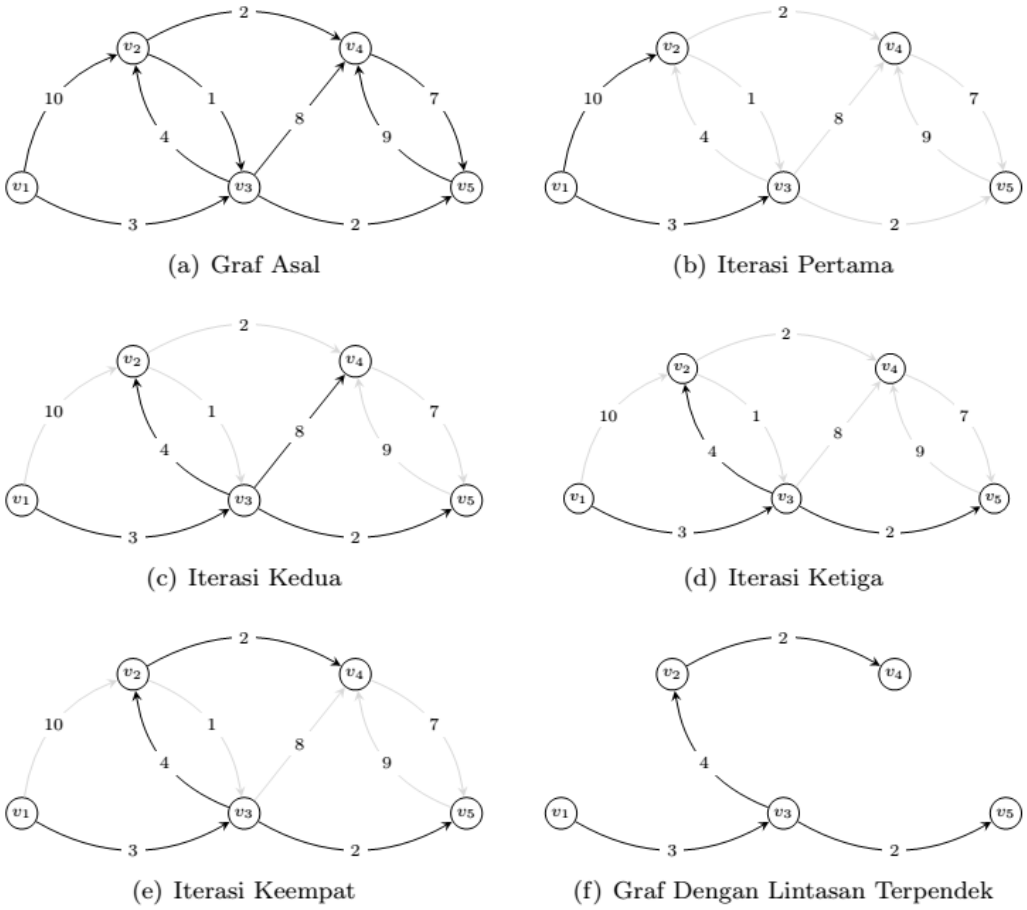
Misalkan $G = (V, E)$ adalah sebuah graf (atau graf berarah) dengan bobot sisi non negatif, dan misalkan $s \in V$ adalah titik awal. Algoritma di atas terdiri dari beberapa langkah yang secara mendasar setiap langkah diperuntukkan untuk masing-masing titik di V. Langkah pertama, menginisialisasi D dengan n buah ∞ dan memberi nilai 0 pada $D[s]$. Tujuan dari pemberian nilai1adalah untuk melambangkan bilangan yang sangat besar. Dimana fungsi D adalah untuk menyimpan jarak semua lintasan terpendek dari titik s ke sembarang titik lainnya di G, dan jarak s ke dirinya sendiri adalah 0. Sedangkan P adalah titik-titik induk yang diinisialisasi kosong dan Q adalah titik-titik yang akan dikunjungi yang diinisialisasi sama dengan semua titik di G. Sehingga sekarang yang dipertimbangkan adalah setiap titik di Q, dan hapus sembarang titik yang telah dikunjungi dari Q. Perulangan *while* pada baris ke-5 terus diproses hingga semua titik telah dikunjungi.

Pada baris ke-6 dipilih titik mana yang akan dikunjungi, misalkan titik yang terpilih adalah v dimana $D[v]$ minimal. Setelah titik v dikunjungi, titik v dihapus dari Q. Perulangan *for* pada baris ke-8 berfungsi untuk menghitung ulang jarak setiap titik u yang bertetangga dengan v, dimana $u \in Q$. Sedangkan kondisi pada baris ke-9 adalah untuk merubah jarak yang telah dihitung ulang. Perintah $D[v] + w(vu)$ adalah jumlah jarak s ke v dan jarak v ke u, dan jika hasil penjumlahannya kurang dari jarak $D[u]$ dari s ke u, maka hasil penjumlahan tersebut dimasukkan ke $D[u]$ dan v menjadi titik induk dari u. Setiap perulangan *while* selesai, jumlah elemen di Q berkurang satu persatu. Pada akhirnya akan dihasilkan D dan P.

Sebagai ilustrasi, algoritma dijkstra akan diterapkan pada Gambar 1(a). Dengan titik awal v_1. Proses algoritma dijkstra pada Gambar 1(a) secara berurutan ditampilkan pada Gambar 1, dengan hasil akhir dari algoritma dijkstra ditampilkan pada Gambar 1(f) dan Tabel 1, dimana setiap 2-tupel bilangan dalam setiap kolom pada Tabel 1 merupakan jarak dan titik induk dari v_i. Dalam setiap iterasi pada algortima dijkstra, jarak dan titik induk selalu diperbaharui, dimana 2-tupel yang digaris bawahi pada setiap iterasinya merupakan titik yang akan dikunjungi. Dari Tabel 1 dapat diperoleh jalur terpendek sebagai berikut:

$$\begin{array}{ll} v_1 - v_2 : v_1, v_3, v_2 & d(v_1, v_2) = 7 \\ v_1 - v_3 : v_1, v_3 & d(v_1, v_3) = 3 \\ v_1 - v_4 : v_1, v_3, v_2, v_4 & d(v_1, v_4) = 9 \\ v_1 - v_5 : v_1, v_3, v_5 & d(v_1, v_5) = 5 \end{array} \quad (1)$$

Sebuah lintasan dari $u - v$ dapat diperoleh dengan proses backtracking dimulai dari v kemudian ke titik induknya dan titik induknya lagi hingga sampai pada u.



GAMBAR 1. ILUSTRASI ALGORITMA DIJKSTRA

TABEL 1. LANGKAH ALGORITMA DIJKSTRA

v_1	v_2	v_3	v_4	v_5
$(\mathbf{0}, -)$	$(\infty, -)$	$(\infty, -)$	$(\infty, -)$	$(\infty, -)$
	$(10, v_1)$	$(\mathbf{3}, \boldsymbol{v_1})$	$(11, v_3)$	$(\mathbf{5}, \boldsymbol{v_3})$
	$(\mathbf{7}, \boldsymbol{v_3})$		$(\mathbf{9}, \boldsymbol{v_2})$	

Sebuah lintasan dari $u - v$ dapat diperoleh dengan proses backtracking dimulai dari v kemudian ke titik induknya dan titik induknya lagi hingga sampai pada u.

B. Pensejajaran Rantai DNA

Sebuah rantai DNA merupakan untaian yang terdiri dari kumpulan empat nukleotida yaitu *adenine*, *cytosine*, *guanine* dan *thymine*. Dimana keempat nukleotida tersebut seraca berturut-turut dilambangkan dengan a; c; g dan t. Sehingga sebuah rantai DNA dapat dituliskan sebagai

$$A = aaattagc \tag{2}$$

Misalkan $\mathcal{A}$ adalah himpunan rantai DNA A dan B, dimana A dan B adalah rantai DNA yang terdiri dari $V_4 = \{a, c, g, t\}$. Tujuan dari pensejajaran adalah untuk memperoleh $\mathcal{A}'$ dengan bobot kemiripan besar atau ketidak miripan kecil dan $\mathcal{A}'$ merupakan himpunan perluasan rantai DNA A dan B yaitu A' dan B'

dengan panjang rantai keduanya sama. Rantai perluasan A' dan B' diperoleh dengan menambahkan "gap" yang biasa dilambangkan dengan virtual simbol "$-$" sehingga A' dan B' terdiri dari $V_5 = \{a, c, g, t, -\}$. Bobot dari $\mathcal{A}'$ ditentukan menggunakan penalti skor atau juga matriks skor, misalnya matriks Hamming

$$d(a,b) = \begin{cases} 0, & \text{jika } a = b \in V_4 \\ 1, & \text{lainnya} \end{cases} \tag{3}$$

Misalkan terdapat dua buah rantai DNA sebagai berikut

$$\begin{cases} A = aaattagc \\ B = gtatatact. \end{cases} \tag{4}$$

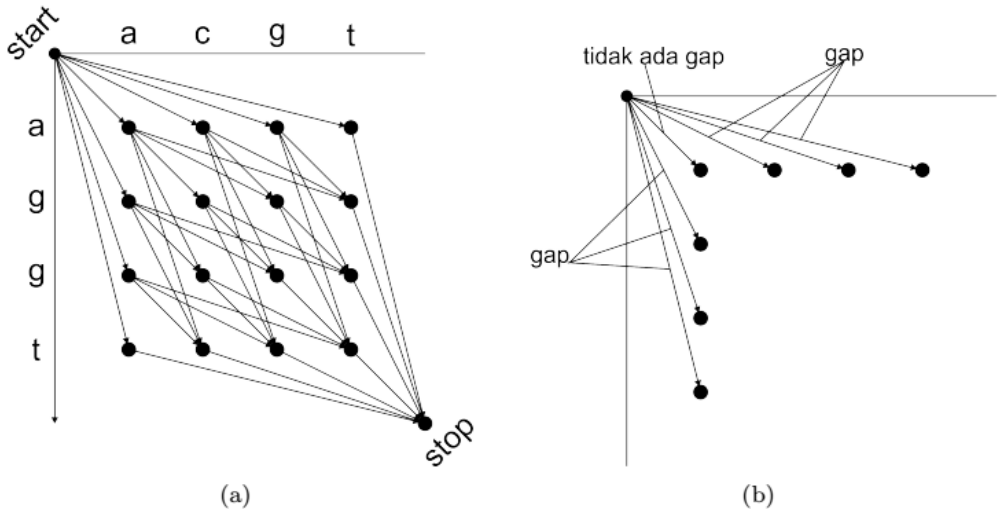
Dengan menggunakan penalti skor 5 jika sesuai, -3 jika tidak sesuai dan -7 jika salah satunya gap. Dengan pemrograman dinamis akan diperoleh

$$\begin{cases} A = aaat - tagc \\ B = gtatatact \end{cases} \tag{5}$$

Dengan bobot minimal sebesar 1.

III. HASIL DAN PEMBAHASAN

Sebelum menerapkan algoritma dijkstra pada pensejajaran rantai DNA, haruslah diperoleh graf yang merepresentasikan hubungan antar rantai DNA dalam pensejajaran sekuens. Graf tersebut terdiri dari titik-titik yang berasal dari semua kemungkinan pasangan nukleotida dari rantai DNA dan ditambah satu titik awal dan satu titik akhir, beberapa titik dihubungkan dengan sisi yang merepresenta sikan gap seperti pada Gambar 2.



GAMBAR 2. GRAF PENSEJAJARAN

Bobot dari tiap sisi sama dengan penalti skor yang bersesuaian dengan pasangan nukleotida yang dituju. Tidak hanya itu saja, keberadaan gap juga perlu dipertimbangkan, sehingga pasangan nukleotida yang tidak mempunyai gap seperti pada Gambar 2(b) mendapat tambahan skor 0. Sedangkan pasangan nukleotida yang mempunyai gap mendapat tambahan kelipatan dari skor gap sesuai dengan urutannya, jadi semakin jauh panjang sisi maka semakin besar kelipat skor gap yang ditambahkan. Penalti skor yang digunakan dibuat sedemikian rupa sehingga pasangan rantai DNA yang sama mendapat skor yang kecil, sedangkan pasangan yang tidak sama mendapat skor yang lebih besar. Sedangkan skor gap yang biasanya digunakan nilai negatif terkecil, pada metode ini skor gap dibuat positif terbesar. Tujuannya agar semakin banyak gap yang didapat, maka semakin besar pula bobot sisi yang menghubungkannya. Karena pada dasarnya prinsip pensejajaran rantai DNA adalah untuk mendapatkan hasil pensejajaran dengan penambahan gap minimal.

Perhatikan kedua rantai DNA berikut

$$\begin{cases} D = act \\ E = at \end{cases} \tag{7}$$

Misalnya penalti skor yang digunakan adalah

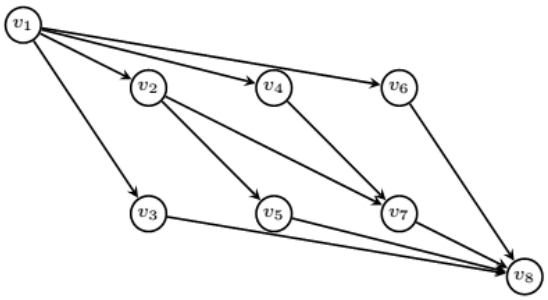
$$d(a,b) = \begin{cases} 0, & \text{jika } a = b \in V_4 \\ 1, & \text{jika } a \neq b \in V_4 \\ 2, & \text{gap} \end{cases} \quad (8)$$

Sehingga didapatkan Tabel 2

TABEL 2. SKOR RANTAI DNA D DAN E

	a	c	t
a	0	1	1
t	1	1	0

Kemudian dapat dibangun graf G dari hubungan rantai DNA D dan E seperti pada Gambar 3. Dimana titik 1 mewakili titik awal, titik 2 mewakili pasangan nukleotida (a,a), titik 3 mewakili pasangan nukleotida(t,a), titik 4 mewakili pasangan nukleotida(a,c), begitu seterusnya hingga titik 8 mewakili titik akhir/berhenti.



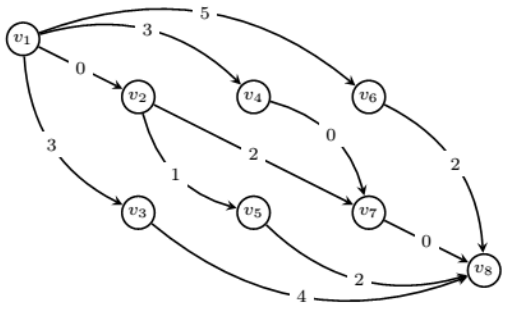
GAMBAR 3. GRAF G

Bobot dari tiap sisi graf pada Gambar 3 disajikan dalam Tabel 3.

TABEL 3. BOBOT SISI GRAF G

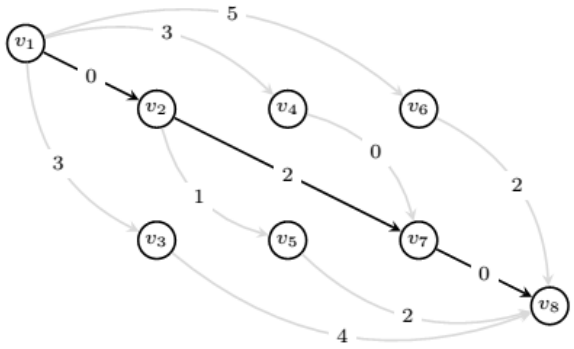
	1	2	3	4	5	6	7	8
1	∞	$0+0=0$	$1+2=3$	$1+2=3$	∞	$1+4=5$	∞	∞
2	∞	∞	∞	∞	$1+0=1$	∞	$0+2=2$	∞
3	∞	∞	∞	∞	∞	∞	∞	$0+4=4$
4	∞	∞	∞	∞	∞	∞	$0+0=0$	∞
5	∞	∞	∞	∞	∞	∞	∞	$0+2=2$
6	∞	∞	∞	∞	∞	∞	∞	$0+2=2$
7	∞	∞	∞	∞	∞	∞	∞	$0+0=0$
8	∞	∞	∞	∞	∞	∞	∞	∞

Kemudian diperoleh graf G dengan bobot-bobot sisinya seperti pada Gambar 4.



GAMBAR 4. GRAF BERBOBOT G

Dengan menerapkan algoritma djikstra pada graf berbobot G, akan diperoleh lintasan seperti pada Gambar 5,



GAMBAR 5. LINTASAN TERPENDEK

yang dapat juga dituliskan dengan $1(\text{start}) \rightarrow 2(a,a) \rightarrow 7(t,t) \rightarrow 8(\text{stop})$. Sehingga diperoleh

$$\begin{cases} D' = act \\ E' = a - t \end{cases} \tag{9}$$

Secara intuitif hasil tersebut adalah himpunan perluasan rantai DNA D dan E dengan skor minimal yang diharapkan, karena gap masuk pada rantai DNA E dan berpasangan dengan nukleotida c pada rantai DNA D.

IV. SIMPULAN DAN SARAN

Algoritma dijkstra adalah algoritma pencarian graf untuk menentukan lintasan terpendek dari titik awal tunggal, tetapi algortim dijkstra juga dapat digunakan untuk mencari lintasan terpendek dari titik awal tunggal dan titik akhir tunggal. Algoritma dijkstra dapat diterapkan dalam pensejajaran sekuens DNA dengan terlebih dahulu membangun graf yang merepresentasikan relasi dalam himpunan Rantai DNA.

Pada penelitian selanjutnya dapat dilakukan dengan mencari metode lain yang lebih baik kinerja daripada aloritma dijkstra.

DAFTAR PUSTAKA

[1] E.W. Dijkstra, "A Note on Two Problem in Connexion with Graphs", Numerische Mathematik, 1, 1959, pp. 269-271.

[2] S.N. Shen and J.A. Tuszynski, Theory and Mathematical Methods for Bioformatics, Springer: Verlag Berlin Heidelberg, 2008.

[3] J. Zhang and Z. Wang, "A New Genetic Algorithm Using Gap Matrixes for Multiple Sequence Alignment", Comm. in Comp. and Inf. Sci. 201, pp. 232-240, 2011.