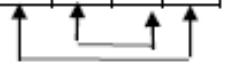
Pembahasan UTS IF2211

Tim Pengajar IF2211 Strategi Algoritma

Soal 1: Brute Foce dan Divide and Conquer

1. (*Inversion problem*) *Netflix* menggunakan sistem rekomendasi untuk merekomendasikan film yang anda sukai. *Netflix* mencoba mencocokkan film kesukaanmu dengan film lainnya. Sistem rekomendasi tersebut adalah sbb: Misalkan kamu me-rangking n buah film. Selanjutnya, *Netflix* memeriksa basisdatanya untuk mencari orang dengan kesukaan film yang mirip. Ukuran kemiripan yang digunakan adalah jumlah inversi antara kedua rangking. Misalkan ranking dari orang tersebut adalah 1, 2, 3, …, n, sedangkan rangking dari kamu adalah a_1, a_2, …, a_n. Film i dan film j disebut inversi jika $i < j$ tetapi $a_i > a_j$. Contoh untuk film A, B, C, D, dan E:

Film	A	B	C	D	E
Ranking saya	1	2	3	4	5
Ranking X	1	3	4	2	5



Inversi: (3, 2) dan (4, 2)

Film	A	B	C	D	E
Ranking saya	1	2	3	4	5
Ranking Y	1	2	4	3	5

Inversi (4, 3)

Karena jumlah inversi dengan Y lebih sedkit daripada X, maka kesukaan saya lebih mirip dengan Y.

Anda diminta menyelesaikan persoalan inversi sbb: Diberikan sebuah senarai A dengan n elemen. Hitunglah jumlah inversi di dalam senarai tersebut. Definisi inversi: Jika $i < j$ tetapi $A[i] > A[j]$ maka pasangan $(A[i], A[j])$ disebut inversi. Contoh: A = [1, 9, 6, 4, 5], maka jumlah inversi adalah 5, yaitu pasangan (9, 6), (9, 4, (9, 5), (6, 4), dan (6, 5).

(a) Hitung jumlah inversi dan pasangan inversinya pada senarai A = [1, 5, 4, 8, 10, 2, 6, 9, 3, 7]. **(7,5)**

Jawaban: ada 17 buah inversi: (5, 4), (4, 2), (8, 2), (10, 2), (6, 3), (9, 3) (5, 2), (4, 3), (8, 6), (10, 6), (9, 7) (5, 3), (8, 3), (10, 9), (10, 3), (8, 7), (10, 7)

(b) Jika persoalan inversi diselesaikan dengan algoritma *Brute-Force*, bagaimana langkah-langkahnya? Jelaskan! Berapa jumlah perbandingan elemen yang dibutuhkan dan berapa kompleksitas algoritma dalam notasi *Big-Oh*?

Jawaban: Mulai dari i = 1...n-1

Mulai dari k = i+1..n

jika A[k] > A[i] maka catat sebagai inversi

$T(n) = (n – 1) + (n – 2) + … + 2 + 1 = n(n – 1)/2 = O(n^2)$

(c) Jika diselesaikan dengan algoritma *Divide and Conquer*, bagaimana langkah-langkahnya? Jelaskan dengan contoh senarai delapan elemen! Berapa jumlah perbandingan elemen yang dibutuhkan dan berapa kompleksitas algoritma dalam notasi *Big-Oh*? Apakah kompleksitas algoritmanya lebih baik dari *brute force*? **(10)**

Jawaban: Bagi larik menjadi dua bagian, lalu untuk setiap bagian hitung inversi.

Tahap combine : bandingkan setiap elemen di larik bagian kiri dengan setiap elemen di bagian kanan. Jika elemen kiri > elemen kanan, catat sebagai inversi. Tahap combine ini membutuhkan perbandingan elemen sebanyak cn^2

$T(n) = 1, \quad n = 2$

$= 2T(n/2) + cn^2, \quad n > 2$

Jika diselesaikan dengan Torema Master, diperoleh $T(n) = O(n^2)$.

Tetap sama dengan Brute Force

(d) Jika digunakan algoritma *Mergesort* dalam menyelesaikan masalah ini, bagaimana caranya? Jelaskan dengan contoh senarai delapan elemen! Berapa kompleksitas algoritmanya? Apakah kompleksitas algoritmanya lebih baik dari jawaban b dan c?

Jawaban: Pada tahap combine dengan Mergesort, masing larik bagian kiri dan larik bagian kanan sudah terurut menaik. Ketika menggabungkan ke matriks B, sekaligus diperiksa apakah elemen bagian kiri lebih besar dari elemen bagian kanan. Jika iya, catat sebagai inversi. Jumlah perbandingan pada tahap combine adalah cn.

$$T(n) = 1, \quad n = 2$$
$$= 2T(n/2) + cn, \quad n > 2$$

Jika diselesaikan dengan Torema Master, diperoleh $T(n) = O(n \log n)$. Lebih baik daripada b dan c

Soal 2: DFS dan BFS

(a) Dari setiap pernyataan di bawah ini, tentukan apakah pernyataan tersebut benar atau salah. Jika benar cukup tuliskan ‘Benar’, jika salah, tuliskan pernyataan yang seharusnya sehingga menjadi pernyataan yang benar.

(i) Cara kerja algoritma DFS seperti struktur data *queue*, dan cara kerja algoritma BFS seperti struktur data *stack*. **(Nilai 3)**

Jawab: Salah

Cara kerja algoritma DFS seperti struktur data *stack*, dan cara kerja algoritma BFS seperti struktur data *queue*.

(ii) Penelusuran secara DFS pada sebuah graf berarah, akan menghasilkan pohon-pohon penelusuran yang sama jika dimulai dari sembarang simpul pada graf tersebut. **(Nilai 3)**

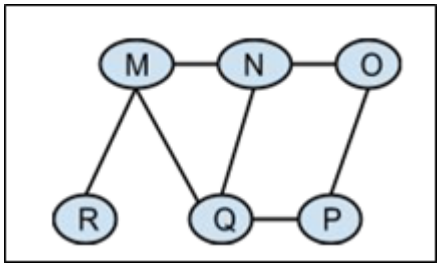
Jawab: Salah

Penelusuran secara DFS pada sebuah graf berarah, akan menghasilkan pohon-pohon penelusuran yang belum tentu sama jika dimulai dari sembarang simpul pada graf tersebut.

(iii) Terdapat graf pada Gambar 1, dan penelusuran secara BFS pada graf tersebut jika dimulai dari simpul Q (dengan *expand* selanjutnya memperhatikan urutan alfabet) adalah QMNPOR. **(Nilai 3)**

Jawab: Salah

Seharusnya: QMNPRO



Soal 2: DFS dan BFS (2)

(b) Terdapat sebuah graf pada Gambar 2, yang menunjukkan keterhubungan antar simpul dan jarak antar simpul.
(i) Tuliskan dengan lengkap tabel yang menunjukkan pohon pencarian jalur dari simpul A ke simpul G dengan cara DFS, dengan urutan pemeriksaan tetangga sesuai urutan alfabet. Tabel berisi informasi simpul ekspan dan simpul hidup. Tuliskan jalur yang dihasilkan dan jarak dari jalur yang dihasilkan. **(Nilai 5)**

Jawab:

Simpul Ekspan	Simpul Hidup
A	B_A,C_A, F_A
B_A	D_{AB}, $J_{AB,}$ C_A, F_A
D_{AB}	E_{ABD}, G_{ABD}, H_{ABD}, $J_{AB,}$ C_A, F_A
E_{ABD}	G_{ABDE}, G_{ABD}, H_{ABD}, $J_{AB,}$ C_A, F_A
G_{ABDE}	Goal ditemukan

Jadi jalur yang ditemukan dengan cara DFS adalah A-B-D-E-G dengan jarak 16.

Soal 2: DFS dan BFS (3)

(b) Terdapat sebuah graf pada Gambar 2, yang menunjukkan keterhubungan antar simpul dan jarak antar simpul.
(ii) Tuliskan dengan lengkap tabel yang menunjukkan pohon pencarian jalur dari simpul A ke simpul G dengan cara BFS, dengan urutan pemeriksaan tetangga sesuai urutan alfabet. Tabel berisi informasi simpul ekspand dan simpul hidup. Tuliskan jalur yang dihasilkan dan jarak dari jalur yang dihasilkan. **(Nilai 5)**

Jawab:
Jadi jalur yang ditemukan dengan cara BFS adalah A-B-D-G dengan jarak 14.

Simpul Ekspan	Simpul Hidup
A	B_A,C_A, F_A
B_A	C_A, F_A, D_{AB}, J_{AB}
C_A	F_A, D_{AB}, J_{AB}, F_{AC}
F_A	D_{AB}, J_{AB}, F_{AC}, H_{AF}, K_{AF}
D_{AB}	J_{AB}, F_{AC}, H_{AF}, $K_{AF,}$ E_{ABD}, G_{ABD}, H_{ABD}
J_{AB}	F_{AC}, H_{AF}, $K_{AF,}$ E_{ABD}, G_{ABD}, H_{ABD}, G_{ABJ}, I_{ABJ}, K_{ABJ}
F_{AC}	H_{AF}, $K_{AF,}$ E_{ABD}, G_{ABD}, H_{ABD}, G_{ABJ}, I_{ABJ}, K_{ABJ}, H_{ACF}, K_{ACF}
H_{AF}	$K_{AF,}$ E_{ABD}, G_{ABD}, H_{ABD}, G_{ABJ}, I_{ABJ}, K_{ABJ}, H_{ACF}, $K_{ACF,}$ I_{AFH}
K_{AF}	E_{ABD}, G_{ABD}, H_{ABD}, G_{ABJ}, I_{ABJ}, K_{ABJ}, H_{ACF}, $K_{ACF,}$ I_{AFH}, J_{AFK}
E_{ABD}	G_{ABD}, H_{ABD}, G_{ABJ}, I_{ABJ}, K_{ABJ}, H_{ACF}, $K_{ACF,}$ I_{AFH}, J_{AFK}, G_{ABDE}
G_{ABD}	Goal ditemukan

Soal 3: Decrease and Conquer

Terdapat sebuah matriks A berukuran , yang sudah terurut menaik elemen-elemennya, sedemikian sehingga untuk ; dan untuk . Persoalan yang akan diselesaikan adalah menentukan apakah sebuah elemen ada pada matriks tersebut. Gunakan pendekatan Decrease and Conquer untuk menyelesaikan persoalan tersebut.

(a) Tuliskan langkah-langkah pendekatan yang anda usulkan, dan tuliskan apakah pendekatan tersebut termasuk *decrease by a contant*, *decrease by factor*, atau *decrease by variable size*. Tentukan juga kompleksitas pendekatan usulan anda dalam notasi Big O. **(Nilai 10)**

Jawab:

Alternatif I:

Penerapan binary search di tiap baris pada matriks. Setiap baris, periksa elemen tengah (e):

(i) jika e<x maka periksa elemen sebelah kanan dari e, dan abaikan elemen kiri dari e;

(ii) Jika e>x maka periksa elemen sebealah kiri dari e, dan abaikan elemen kanan dari e;

(iii) Jika e=x maka elemen yang dicari ditemukan.

Pendekatan ini termasuk *decrease by variable*. Binary search pada tiap baris memerlukan waktu O(log n), dan jika diterapkan pada n baris maka kompleksitas waktunya adalah : O (n log n)

Jika kurang tepat, nilai menjadi 3

Soal 3: Decrease and Conquer (2)

Terdapat sebuah matriks A berukuran , yang sudah terurut menaik elemen-elemennya, sedemikian sehingga untuk ; dan untuk . Persoalan yang akan diselesaikan adalah menentukan apakah sebuah elemen ada pada matriks tersebut. Gunakan pendekatan Decrease and Conquer untuk menyelesaikan persoalan tersebut.

(a) Tuliskan langkah-langkah pendekatan yang anda usulkan, dan tuliskan apakah pendekatan tersebut termasuk *decrease by a contant*, *decrease by factor*, atau *decrease by variable size*. Tentukan juga kompleksitas pendekatan usulan anda dalam notasi Big O. **(Nilai 10)**

Jawab:

Alternatif II:

Pemeriksaan dimulai dari posisi kanan atas ( ), misal nilai elemennya adalah e

(i) Jika e < x, maka seluruh baris pasti lebih kecil dari x, oleh karena itu abaikan seluruh baris, dan periksa baris berikutnya pada kolom yang sama.

(ii) Jika e > x, maka abaikan seluruh kolom tersebut, karena nilai di kolom tersebut pasti lebih besar dari x, periksa kolom berikutnya (1 kolom sebelah kiri) pada baris yang sama.

(iii) Jika e = x maka x ditemukan pada baris dan kolom tersebut.

Untuk setiap langkah, maka seluruh baris atau kolom diabaikan, jadi pendekatannya adalah decrease by a constant (n), dan kompleksitas waktunya adalah O(n).

Soal 3: Decrease and Conquer (3)

(b) Terapkan pendekatan usulan anda (jelaskan langkah-langkahnya) untuk mencari apakah terdapat pada matriks berikut ini, dan hasilkan posisi ditemukannya elemen tersebut. **(Nilai 6)**

$$\begin{bmatrix} 10 & 20 & 30 & 40 \\ 15 & 25 & 35 & 45 \\ 27 & 29 & 37 & 48 \\ 32 & 33 & 39 & 50 \end{bmatrix}$$

Jawab:

Dengan pendekatan Alternatif II:

(i). Periksa elemen [1][4], nilainya adalah 40, dan 40 > 29, maka abaikan seluruh kolom 4, dan bergerak ke 1 kolom di sebelah kiri, yaitu posisi [1][3].

(ii). Nilai pada [1][3] adalah 30, dan 30>29, maka abaikan seluruh kolom 3, dan bergerak 1 kolom ke sebelah kiri, yaitu posisi [1][2].

(iii). Nilai pada posisi [1][2] adalah 20, dan 20 < 29, maka abaikan seluruh baris 1, dan bergerak 1 baris setelahnya, sehingga berada pada posisi [2][2].

(iv). Nilai pada posisi [2][2] adalah 25, dan 25 < 29, maka abaikan seluruh baris 2, dan bergerak ke baris 3 sehingga berada pada posisi [3][2].

(v). Nilai pada posisi [3][2] adalah 29, dan nilai x yang dicari adalah 29. Solusi ditemukan pada posisi [3][2].

Jika jawaban (b) konsisten dengan cara di (a) walau tidak tepat caranya, nilai tetap 6. Jika tidak konsisten nilai menjadi 3.

Soal 4: Exhaustive Search + Greedy

Lakukanlah penjadwalan untuk sebuah mesin yang menerima **7 *job***, dan **setiap *job* diproses selama 1 satuan waktu**. Fungsi objektifnya adalah memaksimumkan profit.

(a) Jelaskanlah bagaimana penjadwalan dilakukan dengan exhaustive search, dan tentukan kompleksitas algoritmanya? **(Nilai 7.5)**

(b) Jelaskanlah strategi greedy untuk penjadwalan ini, dan berikanlah solusinya langkah per langkah. **(Nilai 7.5)**

Job ke-i	1	2	3	4	5	6	7
deadline	2	4	3	2	3	1	1
profit	40	15	60	20	10	45	55

Ji	1	2	3	4	5	6	7
di	2	4	3	2	3	1	1
pi	40	15	60	20	10	45	55

Solusi Soal 4a: *Exhaustive Search*

Tahapan exhaustive search (nilai 5):

1. Enumerasi himpunan bagian (*subset*) *job*
2. Evaluasi kelayakan setiap subset dan hitung profitnya
3. Pilih subset dengan total keuntungan terbesar.

Kompleksitas algoritma: $O(n.2^n)$ (nilai 2.5)

Solusi Soal 4b: Greedy

Ji	1	2	3	4	5	6	7
di	2	4	3	2	3	1	1
pi	40	15	60	20	10	45	55

Strategi greedy: Pada setiap langkah, pilih *job i* yang layak dengan p_i yang terbesar untuk menaikkan nilai fungsi obyektif *F*. (nilai 2.5)

Solusi: {7,1,3,2}

(nilai 5)

Langkah	J	F=sigma(pi)	Keterangan
0	{}	0	-
1	{3}	60	layak
2	{7,3}	115	layak
3	{7,3}+{6}	-	Tidak layak
4	{7,1,3}	155	layak
5	{7,1,3}+{4}	-	Tidak layak
6	{7,1,3,2}	170	layak
7	{7,1,3,2}+{5}	-	Tidak layak

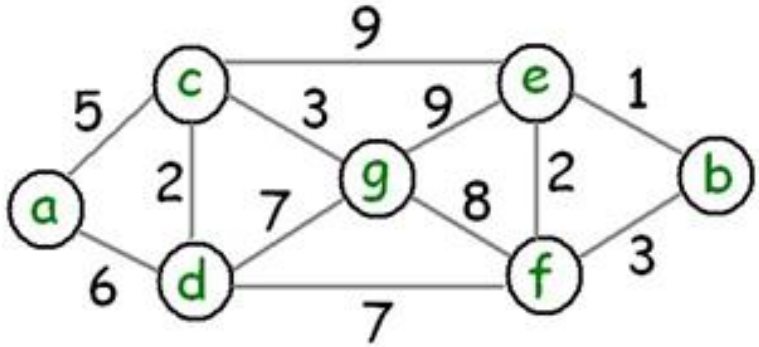
Soal 5: Exhaustive Search + Greedy

Diberikan graf berikut ini, kita akan menentukan **lintasan terpendek dari simpul a ke semua simpul lainnya**.

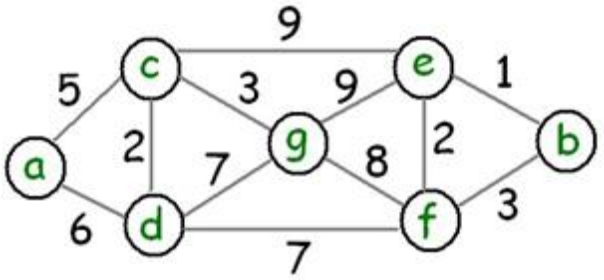
(a) Gunakanlah algoritma Dijkstra untuk menentukan lintasan terpendek tersebut. Sebelum mengerjakan, tuliskanlah strategi greedy yang digunakan Dijkstra. **(Nilai 10)**

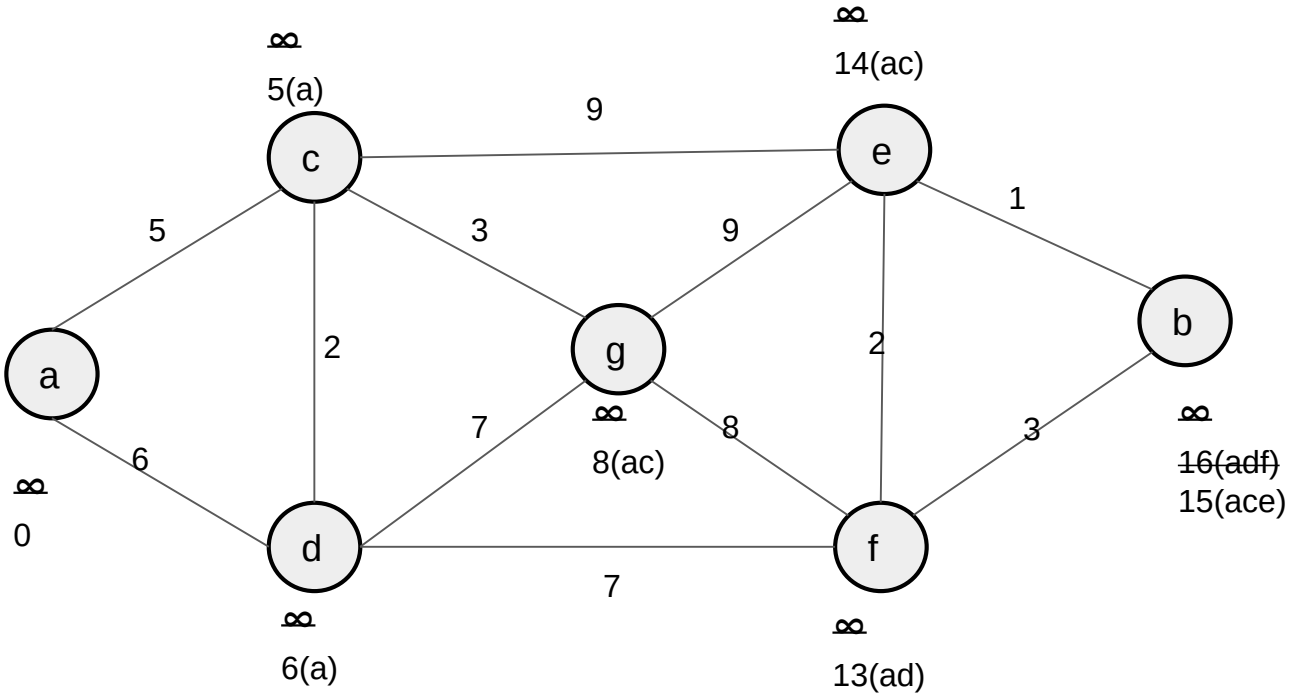
(b) Gunakanlah algoritma Kruskal untuk membentuk pohon merentang minimum, lalu berikanlah lintasan unik dari simpul a ke semua simpul lainnya sebagai lintasan terpendek. Sebelum mengerjakan, tuliskanlah strategi greedy yang digunakan Kruskal. **(Nilai 7.5)**

(c) Bandingkanlah lintasan terpendek dari hasil (a) dan (b), buatlah kesimpulannya. **(Nilai 2.5)**



Solusi Soal 5a: Dijkstra (Nilai 10)





Strategi *greedy*

Pada setiap langkah, pilih simpul yang belum terpilih dan memiliki panjang lintasan terpendek dari simpul awal, lalu update simpul lain yang belum terpilih dengan L(u) + G(u,v) jika L(u)>L(u) + G(u,v).

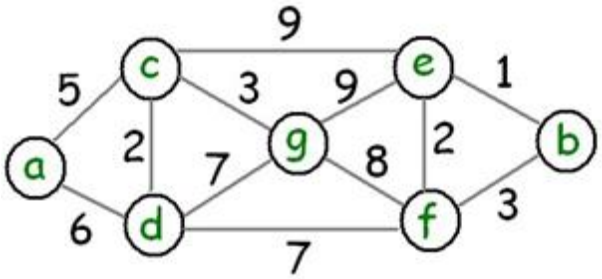
(versi slide kuliah): Pada setiap langkah, ambil sisi yang berbobot minimum yang menghubungkan sebuah simpul yang sudah terpilih dengan sebuah simpul lain yang belum terpilih.

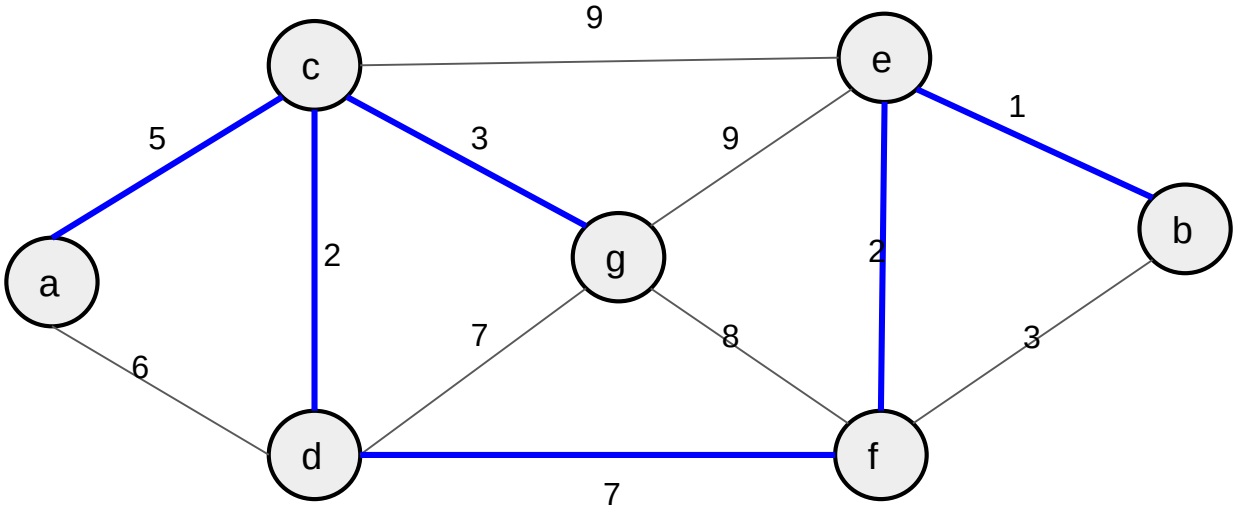
Lintasan terpendek Dijkstra:

a-b: a-c-e-b = 15; a-c: a-c = 5; a-d: a-d = 6

a-e: a-c-e = 14; a-f: a-d-f = 13; a-g: a-c-g = 8

Solusi 5b: Kruskal (nilai 7.5)





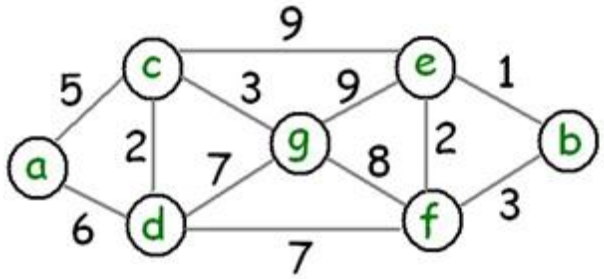
Strategi *greedy*:

Pada setiap langkah, pilih **sisi** *e* dari graf *G* yang mempunyai bobot minimum tetapi *e* tidak membentuk sirkuit di *T*.

Lintasan terpendek:

a-b: a-c-d-f-e-b = 17
a-c: a-c = 5
a-d: a-c-d = 7
A-e: a-c-d-f-e = 16
A-f: a-c-d-f = 14
A-g: a-c-g = 8

Solusi 5c (nilai 2.5)



Lintasan terpendek Dijkstra:
a-b: a-c-e-b = 15; a-c: a-c = 5; a-d: a-d = 6
a-e: a-c-e = 14; a-f: a-d-f = 13; a-g: a-c-g = 8

Lintasan terpendek Kruskal:
a-b: a-c-d-f-e-b = 17; a-c: a-c = 5; a-d: a-c-d = 7
a-e: a-c-d-f-e = 16; a-f: a-c-d-f = 14; a-g: a-c-g = 8

Kesimpulan:
Dijkstra menjamin memberikan lintasan terpendek ke semua simpul; sedangkan Kruskal hanya memberikan pohon merentang minimum yang tidak memberikan lintasan terpendek ke semua simpul.

Soal 1: Brute Force + Divide and Conquer

(*Inversion problem*) *Netflix* menggunakan sistem rekomendasi untuk merekomendasikan film yang anda sukai. *Netflix* mencoba mencocokkan film kesukaanmu dengan film lainnya. Sistem rekomendasi tersebut adalah sbb: Misalkan kamu me-rangking n buah film. Selanjutnya, *Netflix* memeriksa basisdatanya untuk mencari orang dengan kesukaan film yang mirip. Ukuran kemiripan yang digunakan adalah jumlah inversi antara kedua rangking. Misalkan ranking dari orang tersebut adalah 1, 2, 3, …, n, sedangkan rangking dari kamu adalah a_1, a_2, …, a_n. Film i dan film j disebut inversi jika $i < j$ tetapi $a_i > a_j$. Contoh untuk film A, B, C, D, dan E:

Karena jumlah inversi dengan Y lebih sedkit daripada X, maka kesukaan saya lebih mirip dengan Y.

Soal 1: Brute Force + Divide and Conquer

Anda diminta menyelesaikan persoalan inversi sbb: Diberikan sebuah senarai *A* dengan *n* elemen. Hitunglah jumlah inversi di dalam senarai tersebut. Definisi inversi: Jika $i < j$ tetapi $A[i] > A[j]$ maka pasangan $(A[i], A[j])$ disebut inversi. Contoh: A = [1, 9, 6, 4, 5], maka jumlah inversi adalah 5, yaitu pasangan (9, 6), (9, 4, (9, 5), (6, 4), dan (6, 5).

(a) Hitung jumlah inversi dan pasangan inversinya pada senarai A = [1, 5, 4, 8, 10, 2, 6, 9, 3, 7]. **(7,5)**

(b) Jika persoalan inversi diselesaikan dengan algoritma *Brute-Force*, bagaimana langkah-langkahnya? Jelaskan! Berapa jumlah perbandingan elemen yang dibutuhkan dan berapa kompleksitas algoritma dalam notasi *Big-Oh*? **(7,5)**

(c) Jika diselesaikan dengan algoritma *Divide and Conquer*, bagaimana langkah-langkahnya? Jelaskan dengan contoh senarai delapan elemen! Berapa jumlah perbandingan elemen yang dibutuhkan dan berapa kompleksitas algoritma dalam notasi *Big-Oh*? Apakah kompleksitas algoritmanya lebih baik dari *brute force*?

(10)

(d) Jika digunakan algoritma *Mergesort* dalam menyelesaikan masalah ini, bagaimana caranya? Jelaskan dengan contoh senarai delapan elemen! Berapa kompleksitas algoritmanya? Apakah kompleksitas algoritmanya lebih baik dari jawaban b dan c? **(10)**

Soal 2a: DFS and BFS

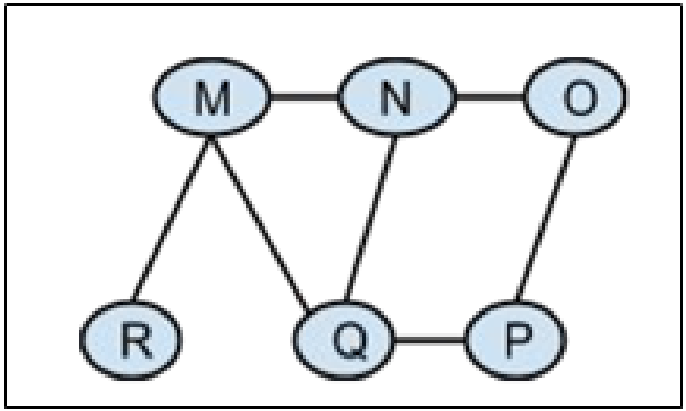
Terdapat dua bagian soal sebagai berikut.

(a) Dari setiap pernyataan di bawah ini, tentukan apakah pernyataan tersebut benar atau salah. Jika benar cukup tuliskan ‘Benar’, jika salah, tuliskan pernyataan yang seharusnya sehingga menjadi pernyataan yang benar.

i. Cara kerja algoritma DFS seperti struktur data *queue*, dan cara kerja algoritma BFS seperti struktur data *stack*. **(Nilai 3)**

ii. Penelusuran secara DFS pada sebuah graf berarah, akan menghasilkan pohon-pohon penelusuran yang sama jika dimulai dari sembarang simpul pada graf tersebut. **(Nilai 3)**

iii. Terdapat graf pada Gambar 1, dan penelusuran secara BFS pada graf tersebut jika dimulai dari simpul Q (dengan *expand* selanjutnya memperhatikan urutan alfabet) adalah QMNPOR. **(Nilai 3)**



Soal 2b

(a) Terdapat sebuah graf pada Gambar 2, yang menunjukkan keterhubungan antar simpul dan jarak antar simpul.

i. Tuliskan dengan lengkap tabel yang menunjukkan pohon pencarian jalur dari simpul A ke simpul G dengan cara DFS, dengan urutan pemeriksaan tetangga sesuai urutan alfabet. Tabel berisi informasi simpul ekspan dan simpul hidup. Tuliskan jalur yang dihasilkan dan jarak dari jalur yang dihasilkan. **(Nilai 5)**

ii. Tuliskan dengan lengkap tabel yang menunjukkan pohon pencarian jalur dari simpul A ke simpul G dengan cara BFS, dengan urutan pemeriksaan tetangga sesuai urutan alfabet. Tabel berisi informasi simpul ekspan dan simpul hidup. Tuliskan jalur yang dihasilkan dan jarak dari jalur yang dihasilkan. **(Nilai 5)**

Soal 3: *Decrease and Conquer*

Terdapat sebuah matriks A berukuran n x n, yang sudah terurut menaik elemen-elemennya, sedemikian sehingga A[i][j]<A[i][j^'] untuk j<j'; dan A[i][j]<A[i']][j] untuk i<i'. Persoalan yang akan diselesaikan adalah menentukan apakah sebuah elemen x ada pada matriks tersebut. Gunakan pendekatan Decrease and Conquer untuk menyelesaikan persoalan tersebut.

Tuliskan langkah-langkah pendekatan yang anda usulkan, dan tuliskan apakah pendekatan tersebut termasuk decrease by a contant, decrease by a constant factor, atau decrease by variable size. Tentukan juga kompleksitas pendekatan usulan anda dalam notasi Big O. (Nilai 10)

Terapkan pendekatan usulan anda (langkah per langkah) untuk mencari apakah x=29 terdapat pada matriks berikut ini, dan hasilkan posisi ditemukannya elemen tersebut. (Nilai 6)