BAB II

MATERI PENUNJANG

2.1 GRAPH BERARAH

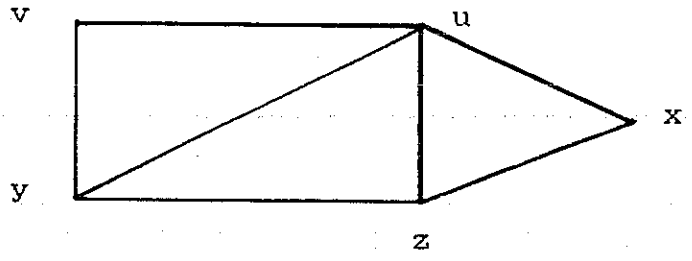
Definisi 2.1

Graph G adalah suatu himpunan tidak kosong dari titik-titik V(G) = $\{v_1, v_2, v_3, \ldots . v_n\}$ beserta himpunan garis-garis E(G) = $\{(v_1,v_2),(v_2,v_3),\ldots..(v_3,v_4)\}$ dan boleh kosong. Graph G dinyatakan oleh G(V,E) dan ruas garis berpangkal dan berujung pada titik.

Titik-titik dari sebuah graph dapat digunakan untuk mengambarkan objek dan garis-garis mengambarkan hubungan diantara objeknya.

Contoh 1.1

Berikut ini suatu graph G(V,E), dengan himpunan titik V = {x,z,y,v,u} dan himpunan garis E = {(x,z),(x,y),(z,y),(y,v),(v,u),(u,z),(u,x)}. Graph ini dapat digambarkan secara geometris seperti berikut ini.



Gambar 2.1 Graph dengan 5 titik dan 7 garis.

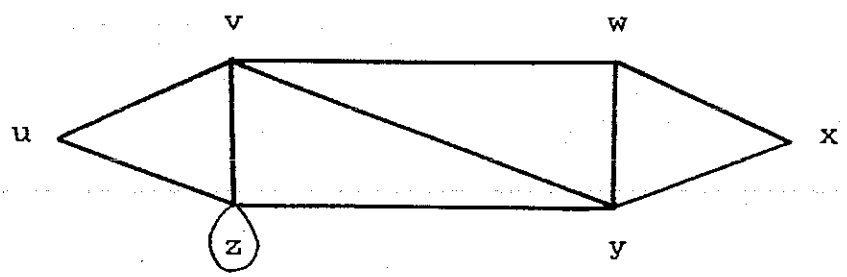
Definisi 2.2

Suatu walk (jalan) yang panjangnya k dalam graph G adalah urutan k garis G yang berbentuk uv,vw,wx,....,yz. Walk ini dinotasikan dengan uvwx...yz,dan disebut walk antara u dan z.

Titik kedua setiap garis adalah sama dengan titik pertama garis berikutnya, maka walk ini dapat dianggap berawal dari u ke v kemudian ke w dan seterusnya, sampai berakhir di titik z. Karena garis-garis itu tidak memiliki arah tertentu, maka walk itu juga dapat dianggap berawal dari z ke y dan seterusnya sampai u. Jadi walk itu juga disebut walk antara z dan u.

Contoh 1.2

Berikut ini walk zuvwxywvzzy dengan panjang 9 antara u dan y, yang memuat garis vw dua kali, titik v,w,y dan z dua kali. walk ini dapat digambarkan secara geometris seperti berikut.



Gambar 2.2. Walk dengan panjang 9 antara u dan y.

Definisi 2.3

Jika semua garis (tetapi tidak perlu semua titik) suatu walk berbeda, maka walk itu disebut trail. Jika semua titiknya juga berbeda, maka trail itu disebut path.

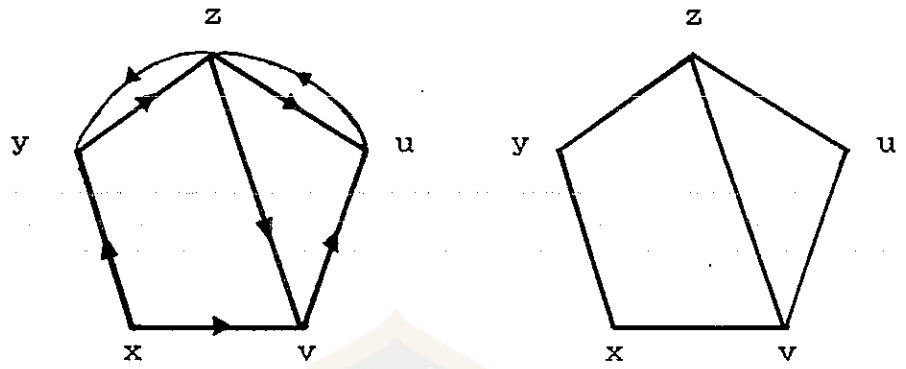
Definisi 2.4

Suatu graph berarah *(digrap)* adalah suatu graph G yang terdiri dari suatu himpunan V yaitu titik dan suatu himpunan E yaitu garis, yang terdiri dari pasangan terurut dari titik yang dituliskan sebagai (x,y) yang disebut garis berarah.

contoh 1.3

Misalkan G(V,E) adalah suatu graph berarah dengan himpunan titiknya adalah V = { x,y,z,u,v } dan garis berarah E = { (x,y), (y,z), (z,y), (z,v), (u,z), (z,u), (v,u),(x,y) }.

Himpunan titik dan garis diatas apabila dibuat gambar terlihat seperti dibawah ini, pada gambar disebelah kiri adalah graph berarah dan disebelah kanan adalah graph tidak berarah.



Gambar 2.3.

Graph berarah dan graph tidak berarah.

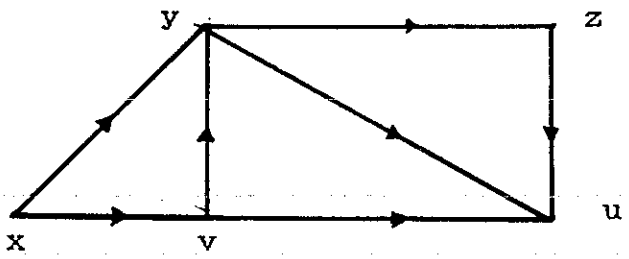
Definisi 2.5

Suatu garis (x,y) dalam suatu graph, titik x disebut titik awal dan titik y disebut titik akhir.

Definisi 2.6

Derajat (*degre*) suatu titik adalah banyaknya garis yang bertemu pada titik tersebut. Dalam suatu graph berarah yang dimaksud dengan derajat masuk (*in degre*) ialah jumlah garis yang masuk pada suatu titik dan yang dimaksud dengan derajat keluar (*out degre*) ialah garis yang keluar dari suatu titik. Maka derajat suatu titik adalah jumlah derajat masuk dan derajat keluar.

Contoh 1.4



Gambar 2.4. digrap dengan 5 titik yang mempunyai derajat masuk dan derajat keluar.

d(x)= 2 d(z) = 2 d(v) = 3

d(y)= 4 d(u) = 3

2.2 MATRIKS

Definisi 2.7

Matriks adalah himpunan skalar (bilangan riil atau komplek) yang disusun (dijajarkan) secara empat persegi panjang (menurut baris dan kolom).

Contoh 1.5

$$A = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$$

Definisi 2.8

Matriks bujur sangkar adalah matriks yang mempunyai jumlah baris dan jumlah kolom sama.

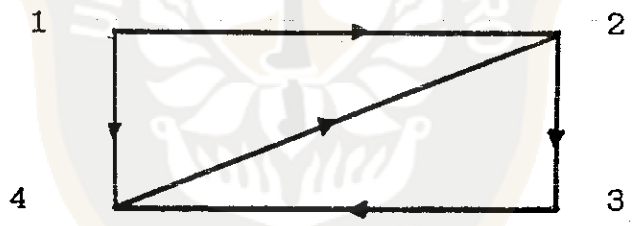
2.3 MATRIKS AJASENSI

Definisi 2.9

Misalkan V = {1,2,3,4....,n}.Matriks ajasensi untuk graph berarah G adalah sebuah matriks A = $(a_{i,j})$ dengan ukuran n x n dan dengan $a_{i,j}$ = 1 atau $a_{i,j}$= 0. $a_{i,j}$ = 1 jika terdapat sebuah garis dari titik i ke titik j, dan $a_{i,j}$= 0 bila tidak ada garis tersebut.

Contoh 1.6

Jika diberikan graph berarah dengan 4 titik .



Gambar 2.5.Graph berarah dengan 4 titik.

Maka matriks ajasensi dari graph G.

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

2.4 PENUTUP TRANSITIF-REFLEKSIF DARI GRAPH BERARAH

Jika A matriks ajasensi dari graph berarah , maka dari A dapat dibentuk matriks $A^*=(a^*_{i,j})$ yang disebut matriks penutup Transitif-Refleksif.

Definisi 2.10

Jika diberikan V = {1,2,3...n.}. Penutup Transitif-Refleksif dari graph berarah G adalah sebuah matriks $A^*= (a^*_{i,j})$ dengan ukuran n x n, dengan $a^*_{i,j}= 1$ atau $a^*_{i,j} = 0$. $a^*_{i,j}= 1$ jika terdapat path dengan panjang ≥ 0 dari i ke j dan $a^*_{i,j}= 0$ jika tidak ada path tersebut.

Contoh 1.7

Diberikan graph berarah G pada gambar 2.6 yang terdiri dari 4 titik.

Gambar 2.6 Graph berarah dengan 4 titik.

Maka matriks A^* yang terjadi

$$A^* = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Pada matriks A^*, $a^*_{11} = a^*_{22} = a^*_{33} = a^*_{44}= 1$ karena terdapat path dengan panjang 0. Sedangkan $a^*_{12}= a^*_{13}= a^*_{14}= 1$ karena terdapat path dari 1 ke 2 dengan panjang 1 dan dari 1 ke 3 dengan panjang 2 dan dari 1 ke 4 dengan panjang 3. Begitu juga untuk yang lain.

2.5 DASAR-DASAR ALGORITMA

Definisi 2.11

Suatu algoritma adalah prosedur terbatas untuk menyelesaikan sebuah permasalahan dengan langkah-langkah tertentu dalam waktu yang terbatas.

Contoh 1.8

Diberikan sebuah algoritma untuk mencari pembagi bersama terbesar. Jika diberikan 2 (dua) bilangan bulat positip m dan n untuk menentukan pembagi bersama terbesar yaitu bilangan bulat positip terbesar yang dapat membagi m dan n.

E_1 [Temukan sisa]. Bagi m dan n, berikan r sebagai sisa pembagian.(dalam hal ini $0 \leq r < n$).

E_2 [Apakah sisanya nol]. Jika $r = 0$, algoritma berakhir, n sebagai jawaban .

E_3 [Menukar tempat]. $m \leftarrow n$, $n \leftarrow r$, dan kembali kelangkah E_1.

Agar algoritma dapat diproses, harus dipenuhi syarat-syarat algoritmanya. Adapun syarat-syarat algoritma tersebut adalah.

1. Mempunyai input.

   Suatu algoritma dapat mempunyai satu atau banyak input, yaitu merupakan suatu kwantitas dimana diberikan nilai awal sebelum algoritma dimulai. Input diperoleh dari himpunan objek yang telah ditetapkan.

Pada algoritma diatas nilai input yang diberikan m dan n, yang keduanya diambil dari himpunan integer positif.

2. Mempunyai output.

   Dari setiap nilai input dari suatu algoritma menghasilkan nilai output dari himpunan yang dispesifikasikan . Nilai output adalah penyelesaian dari permasalahan. Satu algoritma dapat mempunyai satu atau banyak output.

3. Mempunyai kepastian.

   Langkah-langkah sebuah algoritma seharusnya didefinisikan dengan tepat, tindakan penyelesaian harus ditetapkan dengan tepat dan jelas untuk setiap kasus.

4. Mempunyai keterbatasan.

   Sebuah algoritma harusnya menghasilkan output yang diinginkan setelah akhir dari langkah untuk sembarang himpunan input.

5. Mempunyai keefektipan.

   Algoritma harus memungkinkan untuk setiap langkahnya dilakukan pada sebuah algoritma yang sebenarnya dan mempunyai jumlah waktu yang terbatas.Artinya semua operasi yang dilakukan dalam algoritma adalah operasi dasar.

2.6 KOMPLEKSITAS WAKTU

Salah satu alat yang digunakan untuk mengetahui efisiensi sebuah algoritma adalah waktu yang digunakan oleh komputer untuk menyelesaikan sebuah permasalahan dengan menggunakan algoritma tersebut, ketika sebuah nilai input ukurannya telah dispesifikasikan. Sebuah analisa waktu yang diinginkan untuk menyelesaikan sebuah permasalahan pada ukuran tertentu termasuk dalam kompleksitas waktu.

Kompleksitas waktu pada sebuah algoritma dapat diekpresikan pada jumlah operasi yang digunakan oleh algoritma saat input mempunyai ukuran tertentu. Operasi-operasi yang digunakan untuk mengukur kompleksitas waktu dapat berupa perbandingan bilangan bulat, perkalian bilangan bulat, pembagian bilangan bulat ataupun sembarang operasi dasar lainnya. Untuk menunjukkan kompleksitas waktu dari suatu algoritma dikenal notasi "big Oh".

Definisi 2.12

Diberikan f(n) dan g(n) suatu fungsi dari himpunan bilangan bulat atau riil. Dikatakan f(n) adalah O(g(n)), dibaca "f(n) adalah big Oh dari g(n) ", jika dapat ditemukan bilangan positif C dan k sedemikian sehingga

$$| f(n)| \leq C | g(n) |$$

untuk semua n > k.

Suatu fungsi f(n) termasuk dalam himpunan O(g(n)) jika dapat ditemukan bilangan positif C dan k sedemikian sehingga| f(n)| kurang dari C|g(n)| untuk suatu harga n>k. Penulisan f(n) adalah O(g(n)) menunjuhkan bahwa f(n) anggota dari O(g(n)) atau f(n) ∈ O(g(n)).

Contoh 1.9

Diberikan f(n) dan g(n) fungsi dari himpunan bilangan riil, yang didefinisikan oleh $f(n) = n^3+7n^2+11n$ dan $g(n) = n^3$ maka f(n) adalah O(g(n)).

Analisa :

Jika dipilih C = 19 dan berlaku untuk n >1, maka diperoleh.

$$\begin{aligned} |f(n)| &= |n^3+ 7n^2+ 11n| \\ &\le |n^3+ 7n^3+ 11n^3| \\ &= 19n^3 \\ &\le C\,|g(n)| \end{aligned}$$

Dengan demikian f(n) adalah O(g(n)).

Teorema 2.1

Diberikan $f(n) = a_m n^m + a_{m-1} n^{m-1} + \ldots\ldots + a_1 n + a_o$, dimana $a_o, a_1, .., a_m$ adalah bilangan real dan $g(n) = n^m$ Maka f(n) adalah $O(n^m)$.

Bukti:

Pilih $C = |a_m|+|a_{m-1}|+.....+|a_1|+|a_0|$ dan $n>1$ maka

$$
\begin{aligned}
|f(n)| &= |a_m n^m + a_{m-1} n^{m-1} + \ldots + a_1 n + a_0| \\
&\le |a_m n^m| + |a_{m-1} n^{m-1}| + \ldots + |a_1 n| + |a_0| \\
&\le |a_m| n^m + |a_{m-1}| n^{m-1} + \ldots + |a_1| n + |a_0| \\
&\le |a_m| n^m + |a_{m-1}| n^m + \ldots + |a_1| n^m + |a_0| n^m \\
&\le (|a_m|+|a_{m-1}|+.....+|a_1|+|a_0|) n^m \\
&= C.n^m \\
&= C\,|g(n)|
\end{aligned}
$$

Maka terbukti bahwa $f(n)$ adalah $O(g(n))$.

■

Teorema 2.2

Jika $f_1(n)$ adalah $O(g_1(n))$ dan $f_2(n)$adalah $O(g_2(n))$, maka $(f_1(n) + f_2(n))$ adalah $O(\max((g_1(n), g_2(n)))$.

Bukti:

Misalkan $f_1(n)$ adalah $O(g_1(n))$ dan $f_2(n)$ adalah $O(g_2(n))$ dari definisi notasi big Oh terdapat konstanta C_1, C_2. k_1 dan k_2 sedemikian sehingga

$|f_1(n)| \le C_1 |g_1(n)|$, untuk $n > k_1$ dan

$|f_2(n)| \le C_2 |g_2(n)|$, untuk $n > k_2$

dengan menggunakan persamaan segitiga $|a+b| \le |a| + |b|$, untuk n lebih besar dari k_1 dan k_2 dengan berdasar pada pertidaksamaan untuk $|f_1(n)|$ dan $|f_2(n)|$ maka :

$$|f_1(n) + f_2(n)| \le |f_1(n)| + |f_2(n)|$$

$$| f_1(n) | + | f_2(n)| \leq C_1| g_1(n)| + C_2| g_2(n)|$$
$$\leq C_1| g(n)| + C_2| g(n) |$$
$$= (C_1+C_2)| g(n) |$$
$$= C | g(n) |$$

dimana $C = C_1 + C_2$ dan $g(n) = \max (| g_1(n)|,| g_2(n) |)$. $\max(g_1(n),g_2(n))$ menunjukkan maksimum atau lebih besar pada $g_1(n)$ atau $g_2(n)$. Pertidaksamaan tersebut menunjukkan bahwa $|(f_1 + f_2) (n) \leq C | g(n) |$ maka

$(f_1(n) + f_2(n))$ adalah $O(\max((g_1(n) , (g_2(n)))$.

Terbukti. ■

Teorema 2.3

Jika $f_1(n)$ adalah $O(g_1(n))$ dan $f_2(n)$ adalah $O(g_2(n))$, maka $(f_1.f_2)(n)$ adalah $O(g_1(n).g_2(n))$.

bukti:

Misalkan $f_1(n)$ adalah $O(g_1(n))$ dan $f_2(n)$ adalah $O(g_2(n))$, dari definisi notasi big oh terdapat konstanta C_1, C_2. k_1 dan k_2 sedemikian sehingga

$| f_1(n) | \leq C_1| g_1(n) |$, untuk $n > k_1$ dan

$| f_2(n) | \leq C_2| g_2(n) |$, untuk $n > k_2$

Dengan menggunakan pertidaksamaan segitiga big Oh dapat diperoleh dengan perkalian fungsi f_1 dan f_2. Untuk n lebih besar dari pada $\max (k_1,k_2)$.

$$| (f_1.f_2)(n) | = | f_1(n) || f_2(n)|$$
$$\leq C_1 | g_1(n) | C_2| g_2(n) |$$
$$\leq C_1.C_2 | (g_1.g_2) (n) |$$
$$\leq C | (g_1,g_2)(n) |$$

dengan C = $C_1.C_2$. Dari pertidaksamaan tersebut dengan berdasar pada definisi maka $f_1(n).f_2(n)$ adalah $O(g_1.g_2)$. Jika terdapat konstanta C dan k ,yaitu C = $C_1.C_2$ dan k = max (k_1,k_2) maka

$(f_1.f_2)$ (n) adalah O $(g_1(n).g_2(n))$

Terbukti.

■

2.7 LOOP FOR

Kompleksitas waktu dari suatu loop adalah jumlah iterasi(putaran) yang dilakukan untuk menyelesaikan peryataan-peryataan dalam loop. Jika dalam suatu algoritma terdapat loop berkalang, yaitu loop yang berada dalam loop, maka yang menjadi pedoman adalah loop terdalam. Untuk lebih jelasnya jika diberikan loop

for i ⟵ 1 to m do P(i)

maka waktu yang diberikan dalam loop diatas tidak sama dengan perkalian, tetapi hampir sama dengan penjumlahan, penjumlahan tersebut adalah

$$\sum_{i=1}^{m} t(i)$$

2.8 LARIK SISTOLIK (*SYSTOLIC ARRAY*)

Larik sistolik dapat digunakan untuk menghitung perkalian matriks, sehingga dapat pula digunakan untuk menghitung penutup Transitif-Refleksif dari graph berarah G. Prosesor sistolik disingkat prosesor saja, larik prosesor sistolik disingkat dengan larik sistolik.

Definisi 2.13

Larik sistolik adalah suatu larik yang terdiri dari prosesor-prosesor yang hanya dihubungkan pada prosesor yang secara fisik berhubungan dan pengoperasian prosesor-prosesor ini dilakukan secara sinkron.

Pada setiap langkah setiap prosesor mengambil input-input dari kiri,atas dan kanan prosesor, kemudian melakukan komputasi sederhana dan mengeluarkan output-outputnya ke sebelah kanan,bawah dan kiri prosesor atau disimpan dalam prosesor itu sendiri.

Pada dasarnya pengoperasian larik Sistolik ditentukan oleh

1. Kegunaan prosesor-prosesornya.
2. Pola keterhubungan prosesor-prosesornya.
3. Pola kedatangan input-inputnya.

Prosesor Sistolik ada 2 macam, prosesor pertama mengeluarkan output yang kemudian disebut dengan prosesor I . Sedangkan prosesor kedua menyimpan output didalamnya, yang kemudian disebut dengan prosesor II.

2.8.1 LARIK SISTOLIK PROSESOR I

Definisi 2.14

Larik sistolik prosesor I adalah larik sistolik yang pada akhir prosesnya akan mengeluarkan hasilnya dari prosesor.

(a) (b)

Gambar 2.7 Prosesor I dengan 3 masukan dan 3 keluaran.

Pada gambar 2.7 (a) mengambarkan prosesor yang mempunyai 3 arah masukan dan 3 arah keluaran. Masukan pada prosesor adalah a,b dan c. masukan ini dibaca pada awal perputaran proses (*processing cycle*). Maka pada akhir perputaran, prosesor mengeluarkan a dan b sebagai keluaran tanpa perubahan dan menghitung ab, kemudian mengeluarkan c + ab sebagai keluaran ketiga seperti gambar 2.7 (b).

2.8.2 LARIK SISTOLIK PROSESOR II

Definisi 2.15

Larik sistolik prosesor II adalah larik sistolik yang pada akhir prosesnya akan menyimpan hasilnya dalam prosesor.

(a) (b)

Gambar 2.8 Prosesor II dengan 3 masukan dan 3 keluaran.

Pada gambar 2.8 (a) prosesor II yang memilik 2 arah masukan dan 2 arak keluaran. Masukkan pada prosesor adalah operan a,b dan c. Berbeda dengan macam pertama, pada prosesor II ini, operan c tidak datang dari salah satu arah masukan tetapi berada di dalam prosesor. Pada akhir perputaran prosesor mengeluarkan a dan b tanpa perubahan, menghitung ab dan keluaran c+ab tetap didalam prosesor.