TUGAS PERTEMUAN KE-13

STRUKTUR DATA

GRAPH OPERATION

Disusun oleh:

Fauzi Ikhsan fajar Muzaqi	**2022320018**
Rama Pramudya Wibisana	**2022320019**

JURUSAN SISTEM INFORMASI

FAKULTAS INFORMATIKA

UNIVERSITAS BINA INSANI

BEKASI

2023

DAFTAR ISI

KATA PENGANTAR

Puji syukur alhamdulillah kami panjatkan kepada hadirat Tuhan Yang Maha Esa, karena telah melimpahkan rahmat-Nya berupa kesempatan dan pengetahuan sehingga makalah ini bisa selesai pada waktunya. Tujuan penulisan makalah ini adalah untuk menambah pengetahuan dalam pembelajaran matakuliah struktur data, khususnya pada materi *Graph.*

Penulis mengharapkan tugas ini dapat memberikan pengalaman yang berguna baik bagi pembaca, yang tentunya akan menambah ilmu dan wawasan berfikir mahasiswa. Terima kasih juga kami ucapkan kepada teman-teman yang telah berkontribusi dengan memberikan ide-idenya sehingga makalah ini bisa disusun dengan baik dan rapi. Kami berharap semoga makalah ini bisa menambah pengetahuan para pembaca. Namun terlepas dari itu, kami memahami bahwa makalah ini masih jauh dari kata sempurna, sehingga kami sangat mengharapkan kritik serta saran yang bersifat membangun demi terciptanya makalah selanjutnya yang lebih baik lagi.

Penulis juga memohon maaf apabila dalam penulisan makalah ini terdapat kesalahan *Graph* ini terdapat kesalahan pengetikan dan kekeliruan sehingga membingungkan pembaca dalam memahami maksud penulis.

BAB I PENDAHULUAN

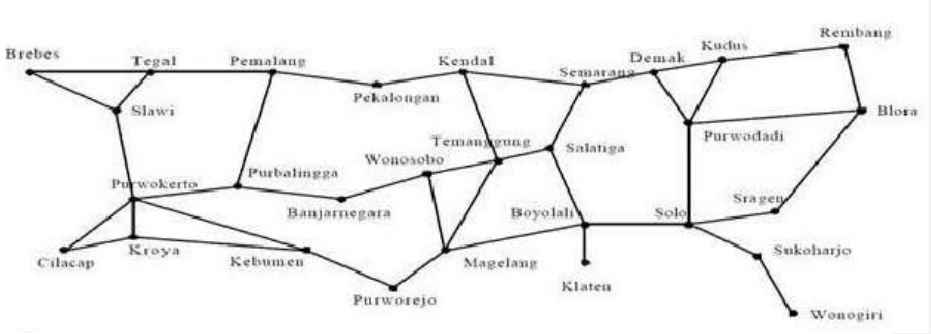
Dalam istilah ilmu komputer, sebuah struktur data adalah cara penyimpanan, pengorganisasian dan pengaturan data di dalam media penyimpanan komputer sehingga data tersebut dapat digunakan secara efisien. Dalam teknik pemrograman, struktur data berarti tata letak data yang berisi kolom-kolom data, baik itu kolom yang tampak oleh pengguna (user) ataupun kolom yang hanya digunakan untuk keperluan pemrograman yang tiadak tampak oleh pengguna.

Graph merupakan struktur data yang paling umum. Jika struktur linear memungkinkan pendefinisian keterhubungan sikuensial antara entitas data, struktur data tree memungkinkan pendefinisian keterhubungan hirarkis, maka struktur graph memungkinkan pendefinisian keterhubungan tak terbatas antara entitas data.

Banyak entitas-entitas data dalam masalah-masalah nyata secara alamiah memiliki keterhubungan langsung (adjacency) secara tak terbatas demikian. Contoh: informasi topologi dan jarak antar kota-kota di pulau Jawa. Dalam masalah ini kota x bisa berhubungan langsung dengan hanya satu atau lima kota lainnya. Untuk memeriksa keterhubungan dan jarak tidak langsung antara dua kota dapat diperoleh berdasarkan data keterhubungan - keterhubungan langsung dari kota-kota lainnya yang memperantarainya. Representasi data dengan struktur data linear ataupun hirarkis pada masalah ini masih bisa digunakan namun akan membutuhkan pencarian-pencarian yang kurang efisien. Struktur data graph secara eksplisit menyatakan keterhubungan ini sehingga pencariannya langsung (straightforward) dilakukan pada strukturnya sendiri.

Graf adalah salah satu jenis struktur data yang terdiri dari titik (vertex) dan garis (edge), dimana dalam graf tersebut, vertex - vertex yang ada dihubungkan oleh edge, hingga menjadi suatu kesatuan yang disebut graf. Sebagai contoh dari pemodelan graf adalah peta kota kota, dimana kota disini sebagai vertex dan jalur yang menghubungkannya berlaku sebagai edge.

Agar lebih jelas, lihatlah gambar dibawah ini



Dalam gambar tersebut, terdapat beberapa kota yang berada dipulau jawa dimana kota - kota tersebut dihubungkan oleh beberapa jalur jalur yang ada. Untuk contoh diatas kita bisa menganggap bawah kota-kota yang ada merupakan vertex, dan jalur-jalur yang menghubungkan kota-kota tersebut sebagai edge. Sehingga secara keseluruhan peta diatas dapat dibuat pemodelannya sebagai sebuah graf.

Ada terdapat beberapa jenis graf yang bisa kita gunakan, yaitu beberapa diantaranya adalah sebagai berikut :

- Graf Berarah adalah graf yang edge-nya memiliki arah, sebagai contoh edge AB menghubungkan vertex A ke B, dimana hubungan vertex B ke A, harus diperoleh dari edge lain, yaitu edge BA, dan jika edge BA tidak ada, maka vertex B ke A tidak memiliki hubungan, meski vertex A ke B memiliki hubungan.
- Graf Tak Berarah adalah graf yang edge-nya tidak memiliki arah, sehingga jika edge AB menghubungkan vertex A ke B, maka secara otomatis juga menghubungkan vertex B ke AGraf Berbobot : adalah suatu graf dimana edge dari graf tersebut memiliki bobot atau nilai tertentu.
- Graf Tidak Berbobot : adalah suatu graf dimana edge dari graf tersebut tidak memiliki bobot atau nilai. Untuk merepresentasikannya dalam pemrograman komputer, graf dapat disusun dari LinkedList yang berada dalam LinkedList.

BAB II PEMBAHASAN

A. Definisi Graph

Suatu graph didefinisikan oleh himpunan verteks dan himpunan sisi (edge). Verteks menyatakan entitas-entitas data dan sisi menyatakan keterhubungan antara verteks. Biasanya untuk suatu graph G digunakan notasi matematis.

G = (V, E)

Dimana :

G = Graph

V = Simpul atau Vertex, atau Node, atau Titik

E = Busur atau Edge, atau arc

V adalah himpunan verteks dan E himpunan sisi yang terdefinisi antara pasangan-pasangan verteks. Sebuah sisi antara verteks x dan y ditulis {x, y}. Suatu graph H = (V1, E1) disebut subgraph dari graph G jika V1 adalah himpunan bagian dari V dan E1 himpunan bagian dari E.

Cara pendefinisian lain untuk graph adalah dengan menggunakan himpunan keterhubungan langsung Vx. Pada setiap verteks x terdefinisi Vx sebagai himpunan dari verteks-verteks yang adjacent dari x. Secara formal:

Vx = {y| (x ,y) -> E}

Dalam digraph didefinisikan juga terminologi-terminologi berikut ini. Predesesor dari suatu verteks x (ditulis Pred(x)) adalah himpunan semua verteks yang adjacent ke x. Suksesor dari verteks x (ditulis Succ(x)) adalah himpunan semua verteks yang adjacent dari x, yaitu adjacenct set di atas.

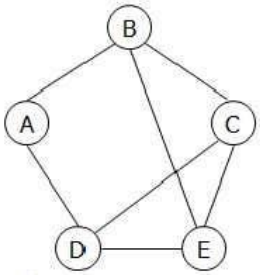
Struktur data yang berbentuk network/jaringan, hubungan antar elemen adalah many-to-many. Contoh dari graph adalah informasi topologi jaringan dan keterhubungan antar kota-kota. Keterhubungan dan jarak tidak langsung antara dua kota sama dengan data keterhubungan langsung dari kota-kota lainnya yang memperantarainya. Penerapan struktur data linear atau hirarkis pada masalah graph dapat dilakukan tetapi kurang efisien. Struktur data graph secara eksplisit menyatakan keterhubungan ini sehingga pencariannya langsung (straight forward) dilakukan pada strukturnya sendiri.

1. Struktur Data Linear = keterhubungan sekuensial antara entitas data .

2. Struktur Data Tree = keterhubungan hirarkis.
3. Struktur Data Graph = keterhubungan tak terbatas antara entitas data

a). Representasi Graph dalam Bentuk Matrik

a. Graph Tak Berarah



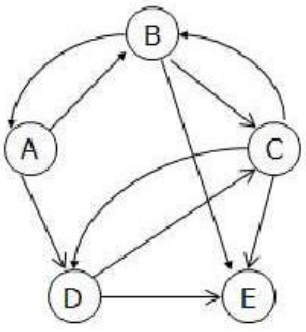
Graf tersebut dapat direpresentasikan dalam sebuah matrik 5x5 , dimana baris dan kolom di matriks tersebut menunjukan vertex yang ada

Urut abiad →

	A 0	B 1	C 2	D 3	E 4
A 0	0	1	0	1	0
B 1	1	0	1	0	1
C 2	0	1	0	1	1
D 3	1	0	1	0	1
E 4	0	1	1	1	0

Degree simpul : 3

b. Graph Berarah



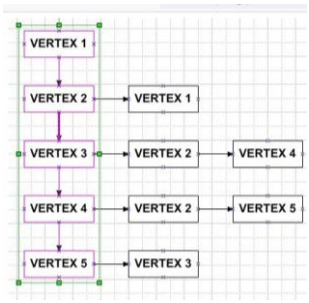
ke → dari ↓

	A 0	B 1	C 2	D 3	E 4
A 0	0	1	0	1	0
B 1	1	0	1	0	1
C 2	0	1	0	1	1
D 3	0	0	1	0	1
E 4	0	0	0	0	0

→ out

Dalam matrik diatas dapat kita lihat bahwa kotak yang berisi angka satu menunjukan bahwa dalam dua vertex tersebut terdapat edge yang menghubungkannya. Dan jika dalam kotak terdapat angka nol, maka hal tersebut menandakan tidak ada edge yang mengubungkan secara langsung dua vertex tersebut.

Untuk representasi dalam pemorgraman komputer, graf tersebut dapat digambarkan seperti dibawah ini :



B. Istilah dalam Graph

1. Incident

   Jika e merupakan busur dengan simpul-simpulnya adalah v dan w yang ditulis e = (v,w), maka v dan w disebut "terletak" pada e, dan e disebut incident dengan v dan w.

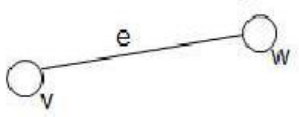
2. Degree

   Di dalam digraph ada yang disebut dengan degree, degree mempunyai 3 jenis antara lain:

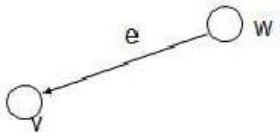
- Degree dari suatu verteks x dalam undigraph adalah jumlah busur yang incident dengan simpul tersebut.
- Indegree dari suatu verteks x dalam digraph adalah jumlah busur yang kepalanya incident dengan simpul tersebut, atau jumlah busur yang "masuk" atau menuju simpul tersebut.
- Outdegree dari suatu verteks x dalam digraph adalah jumlah busur yang ekornya incident dengan simpul tersebut, atau jumlah busur yang "keluar" atau berasal dari simpul tersebut.

3. Adjacent

Pada graph tidak berarah, 2 buah simpul disebut adjacent bila ada busur yang menghubungkan kedua simpul tersebut. Simpul v dan w disebut adjacent.



Pada graph berarah, simpul v disebut adjacent dengan simpul w bila ada busur dari w ke v.

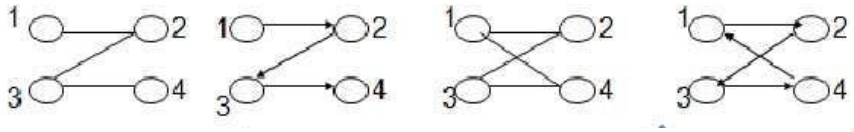


4. Successor dan predecessor

Pada graph berarah, bila simpul v adjacent dengan simpul w, maka simpul v adalah successor simpul w, dan simpul w adalah predecessor dari simpul v.

5. Path

Sebuah path adalah serangkaian simpul-simpul berbeda yang adjacent secara berturut-turut dari simpul satu ke simpul berikutnya.



C. Jenis – Jenis Graph

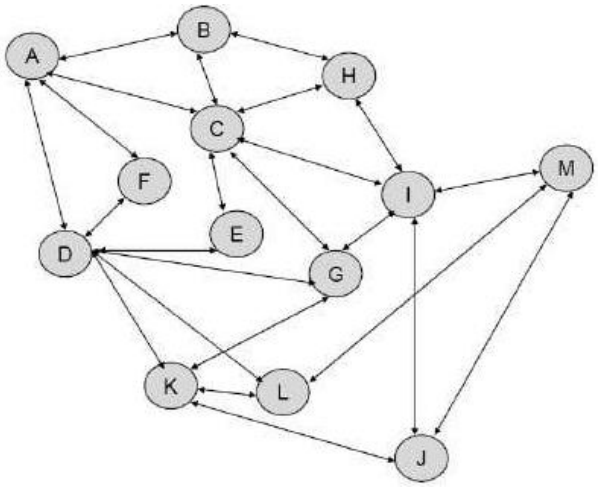
1. Directed Graph (Digraph)

Jika sisi-sisi graph hanya berlaku satu arah. Misalnya : {x,y} yaitu arah x ke y, bukan dari y ke x, x disebut origin dan y disebut terminus. Secara notasi sisi digraph ditulis sebagai vektor (x, y).

Contoh Digraph G = {V, E} :

V = {A, B, C, D, E, F, G, H, I,J, K, L, M}

E = {(A,B), (A,C), (A,D), (A,F), (B,C), (B,H), (C,E), (C,G), (C,H), (C,I), (D,E), (D,F), (D,G), (D,K), (D,L), (E,F), (G,I), (G,K), (H,I), (I,J), (I,M), (J,K), (J,M), (L,K), (L,M)}.

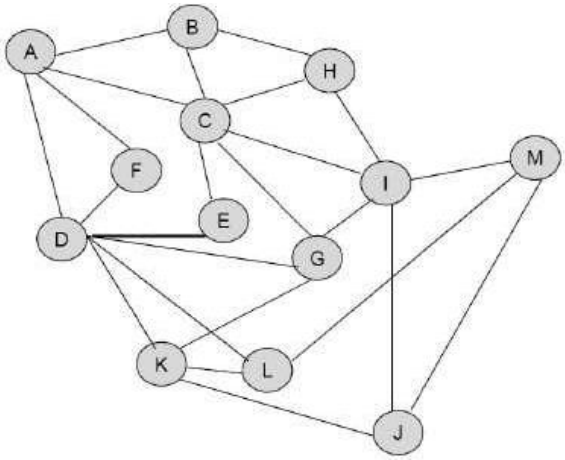


2. Graph Tak Berarah (Undirected Graph atau Undigraph)

Setiap sisi {x, y} berlaku pada kedua arah: baik x ke y maupun y ke x. Secara grafis sisi pada undigraph tidak memiliki mata panah dan secara notasional menggunakan kurung kurawal.

Contoh Undigraph G = {V, E}

V = {A, B, C, D, E, F, G, H, I,J, K, L, M}
E = { {A,B},{A,C}, {A,D}, {A,F}, {B,C}, {B,H}, {C,E}, {C,G}, {C,H}, {C,I}, {D,E}, {D,F}, {D,G}, {D,K}, {D,L}, {E,F}, {G,I}, {G,K}, {H,I}, {I,J}, {I,M}, {J,K}, {J,M}, {L,K}, {L,M}}.



Khusus graph, undigraph bisa sebagai digraph (panah di kedua ujung edge berlawanan) Struktur data linear maupun hirarkis adalah juga graph. Node-node pada struktur linear ataupun hirarkis adalah verteks-verteks dalam pengertian graph dengan sisi-sisinya menyusun node-node tersebut secara linear atau hirarkis. Struktur data linear adalah juga tree dengan pencabangan pada setiap node hanya satu atau tidak ada. Linear 1-way linked list (digraph), linear 2- way linked list (undigraph).

D. Shortest Path

Pencarian shortest path (lintasan terpendek) adalah masalah umum dalam suatu weighted, connected graph. Misal : Pencarian jaringan jalan raya yang menghubungkan kota-kota disuatu wilayah.

- Lintasan terpendek yag menghubungkan antara dua kota berlainan tertentu (Single-source Single-destination Shortest Path Problems).
- Semua lintasan terpendek masing-masing dari suatu kota ke setiap kota lainnya (Single-source Shortest Path problems).
- Semua lintasan terpendek masing-masing antara tiap kemungkinan pasang kota yang berbeda (All-pairs Shortest Path Problems).

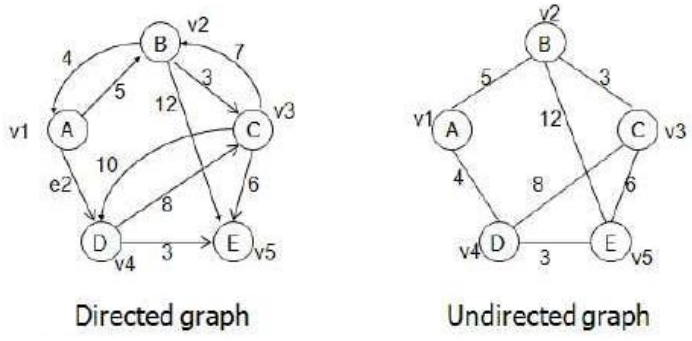
Untuk memecahkan masing-masing dari masalah-masalah tersebut terdapat sejumlah solusi

Dalam beberapa masalah graph lain, suatu graph dapat memiliki bobot negatif dan kasus ini dipecahkan oleh algoritma Bellman-Ford. Yang akan dibahas di sini adalah algoritma Dijkstra yaitu mencari lintasan terpendek dari suatu verteks asal tertentu vs ke setiap verteks lainnya .

1. Graph berbobot (weighted graph)

Apabila sisi-sisi pada graph disertai juga dengan suatu (atau beberapa) harga yang menyatakan secara unik kondisi keterhubungan tersebut maka graph tersebut disebut graph berbobot. Biasanya dalam masalah-masalah graph bobot tersebut merupakan "harga" dari keterhubungan antar vertex. Pengertian "harga" ini menggeneralisasikan banyak aspek, biaya ekonomis dari proses/aktifitas, jarak geografis/tempuh, waktu tempuh, tingkat kesulitan, dan lain sebagainya.

Dalam beberapa masalah lain bisa juga bobot tersebut memiliki pengertian "laba" yang berarti kebalikan dari "biaya" di atas. Dalam pembahasan algoritma-algoritma graph nanti pengertian bobot akan menggunakan pengertian biaya sehingga apabila diaplikasikan pada masalah yang berpengertian laba maka kuantitas-kuantitas terkait adalah kebalikannnya. Misalnya mencari jarak tempuh minimum digantikan dengan mencari laba maksimum.



2. Algoritma Dijkstra’s

Algoritma Dijkstra's :

- Menyelesaikan problem single-source shortest-path ketika semua edge memiliki bobot tidak negatif.
- Algoritma greedy mirip ke algoritma Prim's.

- Algoritma di awali pada vertex sumber s, kemudian berkembang membentuk sebuah tree T, pada akhirnya periode semua vertex dijangkau dari S. Vertex di tambah ke T sesuai urutan.
  Misalnya : Pertama S, kemudian vertex yang tepat ke S, kemudian yang tepat berikutnya dan seterusnya.

3. Dynamic Programming

   Terdiri dari sederetan tahapan keputusan. Pada setiap tahapan berlaku prinsip optimality (apapun keadaan awal dan keputusan yang diambil, keputusan berikutnya harus memberikan hasil yang optimal dengan melihat hasil keputusan sebelumnya.

   Misalnya : Multistage Graph
   Dimana : Cost (i,j) = Min(C(j,l) + Cost(i+1,l)}
   Dengan : C(j,l) = Bobot edge j dan l
   l = Elemen Vi+1 Dan <j,l> eemen E
   i=stage ke-I dan j = node dalam V

Proses dimulai dari k-2, dimana k adalah banyak stage

BAB III PENUTUP

A. KESIMPULAN

Terkait pembahasan diatas dapat disimpulkan bahwa graph adalah oleh himpunan verteks dan himpunan sisi (edge). Representasi graph terbagi dua yaitu graph tak berarah dan graph berarah.

Di dalam graph mempunyai istilah yaitu incident, degree, adjacent dan successor dan predecessor serta path.

B. SARAN

Dengan paparan materi diatas kami menyarankan untuk mempelajarinya lebih dalam. Penulis juga mengharapkan kritik dan saran dalam penulisan makalah graph yang telah kami buat.

Commented [hh1]:

BAB IV DAFTAR PUSTAKA

Makalah graph dibuat oleh fandy dan tim (2010),
https://www.academia.edu/5508323/Makalah_graph