Laporan Praktikum Struktur Data

Semester Gasal 2021/2022

NIM	**71200634**
Nama Lengkap	**Egi Granaldi GInting**
Minggu ke / Materi	**15 / Graph**

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA
2021

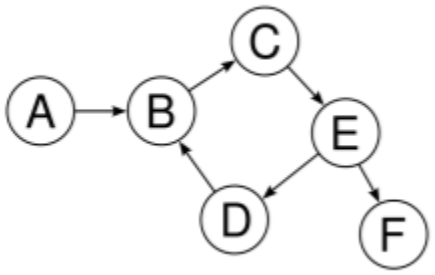
BAGIAN 1: GRAPH

Graph merupakan struktur data yang paling umum.Struktur graph memungkinkan pendefenisian keterhubungan tak terbatas antara entitas data.

Banyak entitas – entititas data dalam masalah – masalah yata secara alamiah memiliki keterhubungan langsung (adjacency) secara tak terbatas demikian.Contohnya, 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.

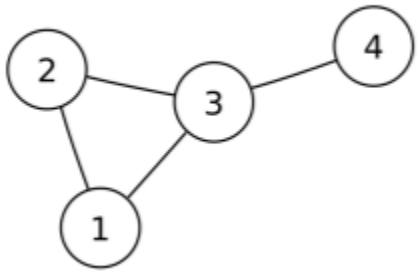
Graph merupakan salah satu jenis struktur data non-linear seperti halnya Tree, di mana Graph terdiri dari dua jenis komponen yaitu edge dan vertex.Vertex merupakan node-node yang menyimpan sebuah nilai, sedangkan edge merupakan garis yang menghubungkan dua buah vertex. Perbedaan yang paling mendasar antara Graph dan Tree adalah pada Tree terdapat hirarki,artinya relasi antara satu node dengan node lain digambarkan dalam hubungan parent dan child, sehingga dapat dikatakan bahwa sebuah node memiliki kedudukan yang lebih tinggi (parent), lebih rendah (child), atau setara (sibling) dengan node yang lain. Sedangkan pada graph, kedudukan setiap node (vertex) adalah setara, sehingga tidak ada node yang memiliki kedudukan lebih tinggi atau lebih rendah dibanding node yang lain.Terdapat beberapa jenis graph, diantarnya adalah graph berarah, graph tidak berarah, dan graph berbobot.

- Graph Berarah
  Graph berarah merupakan graph yang setiap edge-nya memiliki arah, misalnya jika dari vertex A terdapat edge satu arah untuk menuju vertex B, maka dari vertex B kita tidak bisa pergi menuju vertex A.



Gambar 1 Graph berarah

- Graph Tidak berarah
  Graph tidak berarah adalah apabila dari vertex A terdapat edge menuju vertex B, maka berlaku sebaliknya dari vertex B dapat pergi menuju vertex A. Dengan kata lain, graph tidak berarah adalah graph yang seluruh edge-nya memiliki dua arah.

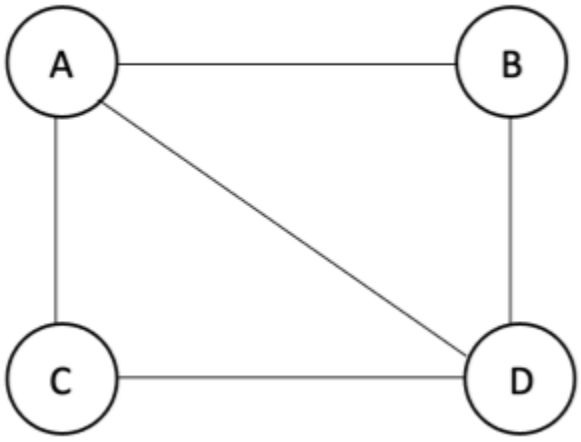


Gambar 2 Graph tidak berarah

- Graph Berbobot
  Graph berbobot memiliki arti setiap edge pada graph memiliki nilai bobot masing-masing untuk mendadakan jarak atau beban yang harus ditempuh saat melakukan perjalanan dari vertex A menuju vertex B.

Implementasi Graph

Graph dapat diimplementasikan menggunakan kombinasi antara list dan dictionary, dengan asumsi setiap vertex memiliki nilai yang unik. Misalnya pada contoh graph tidak berarah dibawah ini.



Gambar 3 Contoh graph tidak berarah

Jika terdapat 4 buah vertex yaitu: a, b, c, d dan terdapat 5 buah edge yaitu : ab, ac, ad, bd, cd. Maka kita dapat membuat struktur dictionary-nya menjadi seperti berikut ini:

```
graph = {
    "a": ["b", "c", "d"],
    "b": ["a", "d"],
    "c": ["a", "d"],
    "d": ["a", "b"]
}
```

Gambar 4 Struktur Dictionary pada graph

Masalah – masalah pada Graph

- Masalah path minimum (Shortest path problem)
  Mencari route dengan jarak terpendek dalam suatu jaringan transportasi.
- Masalah aliran maksimum (maximum flow problem)
  Menghitung volume aliran BBM dari suatu reservoir ke suatu titik tujuan melalui jaringan pipa.
- Masalah pencariah dalam graph (graph searching problem)
  Mencari langkah-langkah terbaik dalam program permainan catur komputer.
- Masalah pengurutan topologis (topological ordering problem)
  Menentukan urutan pengambilan mata-mata kuliah yang saling berkaitan dalam hubungan prasyarat (prerequisite).
- Masalah jaringan tugas (Task Network Problem)
  Membuat penjadwalan pengerjaan suatu proyek yang memungkinkan waktu penyelesaian tersingkat.
- Masalah pencarian pohon rentang minimum (Minimum Spanning Tree Problem)
  Mencari rentangan kabel listrik yang totalnya adalah minimal untuk menghubungkan sejumlah kota.
- Travelling Salesperson Problem
  Tukang pos mencari lintasan terpendek melalui semua alamat penerima pos tanpa harus mendatangi suatu tempat lebih dari satu kali.
- Four-color problem
  Dalam menggambar peta, memberikan warna yang berbeda pada setiap propinsi yang saling bersebelahan.

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). 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.

Diagraph & Undigraph

Graph Berarah (directed graph atau digraph): jika sisi-sisi pada graph, misalnya {x, y} hanya berlaku pada arah-arah tertentu saja, yaitu dari x ke y tapi tidak dari y ke x; verteks x disebut origin dan vertex y disebut terminus dari sisi tersebut. Secara grafis maka penggambaran arah sisi-sisi digraph dinyatakan dengan anak panah yang mengarah ke verteks terminus, secara notasional sisi graph berarah ditulis sebagai vektor dengan (x, y).Graph di samping ini adalah suatu contoh Digraph G = {V, E} dengan V = {A, B, C, D, E, F, G, H, I,J, K, L, M} dan 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)}.

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.Graph di samping ini adalah suatu contoh Undigraph G = {V, E} dengan V = {A, B, C, D, E, F, G, H, I,J, K, L, M} dan 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}}.

Dalam masalah-masalah graph undigraph bisa dipandang sebagai suatu digraph dengan mengganti setiap sisi tak berarahnya dengan dua sisi untuk masing-masing arah yang berlawanan. Undigraph di atas tersebut bisa dipandang sebagai Digraph G = {V, E} dengan V = {A, B, C, D, E, F, G, H, I,J, K, L, M} dan 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), (B,A), (C,A), (D,A), (F,A), (C,B), (H,B), (E,C), (G,C), (H,C), (I,C), (E,D), (F,D), (G,D), (K,D), (L,D), (F,E), (I,G), (K,G), (I,H), (J,I), (M,I), (K,J), (M,J), (K,L), (M,L)}.

Selain itu, berdasarkan definisi ini maka struktur data linear maupun hirarkis adalah juga graph. Node-node pada struktur linear atupun hirarkis adalah verteks-verteks dalam pengertian graph dengan sisi-sisinya menyusun node-node tersebut secara linear atau hirarkis. Sementara kita telah ketahui bahwa struktur data linear adalah juga tree dengan pencabangan pada setiap node hanya satu atau tidak ada. Linear 1-way linked list adalah digraph, linear 2-way linked list bisa disebut undigraph.

Konektivitas pada Unigraph

- Adjacency : Dua verteks dan x dan y yang berlainan disebut berhubungan langsung (adjacent) jika terdapat sisi {x,y} dalam E.
- Path : Sederetan verteks yang mana setiap verteks adjacent dengan verteks yang tepat berada disebelahnya.

- Panjang dari path: jumlah sisi yang dilalui path.
- Siklus: Suatu path dengan panjang lebih dari satu yang dimulai dan berakhir pada suatu verteks yang sama.
- Siklus sederhana: Dalam undigraph, siklus yang terbentuk pada tiga atau lebih verteks-verteks yang berlainan yang mana tidak ada verteks yang dikunjungi lebih dari satu kali kecuali verteks awal/akhir.
- Dua verteks x dan y yang berbeda dalam suatu undigraph disebut berkoneksi (connected) apabila jika terdapat path yang menghubungkannya.
- Himpunan bagian verteks S disebut terkoneksi (connected) apabila dari setiap verteks x dalam S terdapat path ke setiap verteks y (y bukan x) dalam S.
- Suatu komponen terkoneksi (connected components) adalah subgraph (bagian dari graph) yang berisikan satu himpunan bagian verteks yang berkoneksi.
- Suatu undigraph dapat terbagi atas beberapa komponen yang terkoneksi; jika terdapat lebih dari satu komponen terkoneksi maka tidak terdapat path dari suatu verteks dalam satu komponen verteks di komponen lainnya.
- Pohon bebas (free tree): suatu undigraph yang hanya terdapat satu komponen terkoneksi serta tidak memiliki siklus sederhana.

Konektivitas pada Diagraph

- Adjacency ke / dari: Jika terdapat sisi (x,y) maka dalam digraph dikatakan bahwa x "adjacent ke" y atau y "adjacent dari" x. Demikian pula jika terdapat path dari x ke y maka belum tentu ada path dari y ke x Jadi dalam digraph keterkoneksian didefinisikan lebih lanjut lagi sebagai berikut.
- Terkoneksi dengan kuat: Himpunan bagian verteks S dikatakan terkoneksi dengan kuat (strongly connected) bila setiap pasangan verteks berbeda x dan y dalam S, x berkoneksi dengan y dan y berkoneksi dengan x (dpl., ada path dari x ke y dan sebaliknya dari y ke x).
- Terkoneksi dengan Lemah: Himpunan bagian verteks S dikatakan terkoneksi dengan lemah (weakly connected) bila setiap pasangan verteks berbeda x dan y dalam S, salah satu: x berkoneksi dengan y (atau y berkoneksi dengan x) dan tidak kebalikan arahnya (dpl., hanya terdefinisi satu path: dari x ke y atau sebaliknya dari y ke x).

Degree

- Degree dari suatu verteks x dalam undigraph adalah jumlah sisi di mana di salah satu ujungnya terdapat x.
- Indegree dari suatu verteks x dalam digraph adalah jumlah dari predesesor x.
- Outdegree dari suatu verteks x dalam digraph adalah jumlah dari suksesor x.

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 "biaya" dari keterhubungan. Pengertian "biaya" 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.

DAFTAR PUSTAKA

http://aren.cs.ui.ac.id/sda/archive/1998/handout/handout19.html