JARINGAN KOMPUTER LANJUT 2020/2021

TUGAS MINGGU KE - 2

KESIMPULAN VIDEO PEMBELAJARAN MANDIRI

Nama	: Muhammad Fikri Haekal
Nrp	: 15-2018-085
Kelas	: CC
Tanggal Penugasan	: 22 Februari 2021
Tanggal Pengumpulan	: 22 Februari 2021

PROGRAM STUDI INFORMATIKA

INSTITUT TEKNOLOGI NASIONAL

BANDUNG

2021

KESIMPULAN VIDEO PEMBELAJARAN MANDIRI

1. Teori Graf

a. Pengertian / Definisi

Graf adalah himpunan tak kosong berhingga, yang terdiri dari himpunan rusuk dan himpunan simpul yang himpunan simpulnya tidak boleh kosong. Notasi graf: G(V,E) artinya graf G memiliki V simpul dan E rusuk. Simpul-simpul pada graf dapat merupakan obyek sembarang seperti kota, nama orang, jenis buah, komponen alat elektronik dan sebagainya. Rusuk dapat menunjukkan hubungan sembarang seperti rute penerbangan, jalan raya, sambungan telepon, ikatan kimia, dan lain-lain.

Graf juga bisa di sebut Garis yang hanya berhubungan dengan satu titik ujung disebut loop . Dua garis berbeda yang menghubungkan titik yang sama disebut garis paralel. Dua titik dikatakan berhubungan (adjacent) jika garis menghubungkan keduanya. Titik yang tidak memiliki garis yang berhubungan dengannya disebut titik terasing (isolating point). Graf yang tidak memiliki titik (sehingga tidak mewakili garis) disebut garis kosong.

Jika semua garisnya berarah, maka grafnya disebut graf berarah (directed graph), atau sering disingkat di graph . Jika semua garisnya tidak berarah, maka grafnya disebut graf tak berarah (undirected graph). Sehinga dapat ditinjau dari arahnya, graf dapat dibagi menjadi dua yaitu graf berarah dan graf tidak berarah.

Struktur Data Tree merupakan bagian dari Graph, tetapi tidak boleh node-nodenya tidak boleh terhubung sehingga membentuk sirkuit

b. Orde dan Ukuran

banyaknya titik pada graf G disebut *orde*, sebangkan banyaknya sisi pada G disebut ukuran (size). Suatu graf dikatakan hingga jika himpunan titik maulin himpunan sisinya hingga suatu graf dengan hanya satu titik bisa di katakan trivial dan juga nontrivial

c. Adjecent – Incident

Jika ada kedua titik x dan y di titik G maka x bertengangga dengan y , ketika terdapat e adalah suatu sisi diantara x dan y maka x dan y titik ujung dari e, titik x dapat di katakan terkait (incident) dengan suatu sisi e jika x adalah titi ujung dari e kebalikannya jika e di katakan terkait (incident) dengan suatu di titik x maka x dikatakan suatu ujung titik dari e

d. Loop, Link, Graf Sederhana

Suatu susu dengan titik-titik ujung yang identik/sama dikatakan loop. Suatu susu dengan titik-titik ujung yang berbeda disebut link. Suatu graf dapat dikatakan sederhana jika tidak memuat loop dan tidak ada dua link yang menhubungkan pasangan titik yang sama.

e. Derajat

Derajat dari suatu tutu c adalah banyaknya sisi yang terkait dengan v, derajat dari titik v dinotasikan dengan deg(v) sehingga di suatu tutuk dengan derajat 0 dapat dikatakan titik terisolasi (isolation vertex), disebut derajat terkecil pada graf G dinotasikan dengan sigma(G), sedangkat derajat terbesar adalah di notasikan dengan (G)

f. Teorema Graf

Graf G memiliki Sirkuit Euler bila dan hanya bila G adalah graf yang terhubung dan semua titik dalam G mempunyai derajat genap.

2. Algoritma Gready

a. Pengertian / Definisi

Algoritma greedy adalah algoritma yang untuk membentuk solusi langkah langkah perlangkah. Pada setiap langkah tersebut akan dipilih keputusan yang paling optimal. Keputusan tersebut tidak perlu memperhatikan keputusan selanjutnya yang akan diambil, dan keputusan tersebut tidak dapat diubah lagi pada langkah selanjutnya.

Pada algoritma ini , membentuk solusi langkah per langkah (step by step). Terdapat banyak pilihan yang perlu dieksplorasi pada setiap langkah solusi. Oleh karena itu, pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan. Keputusan yang telah diambil pada suatu langkah tidak dapat diubah lagi pada langkah selanjutnya.

b. Prinsip dari Algoritma Greedy

Prinsip utama algoritma greedy adalah “take what you can get now”. Maksud dari prinsip tersebut adalah pada setiap langkah dalam algoritma greedy, diambil keputusan yang paling optimal untuk langkah tersebut tanpa memperhatikan konsekuensi pada langkah selanjutnya. Solusi tersebu disebut dengan optimum lokal. Kemudian saat pengambilan nilai optimum lokal pada setiap langkah, diharapkan tercapai optimum gloal, yaitu tercapainya solusi optimum yang melibatkan keseluruhan langkah dari awal sampai akhir.

c. Elemen Algoritma Greedy

1. Himpunan kandidat, *C*.
2. Himpunan solusi, *S*
3. Fungsi seleksi (*selection function*)
4. Fungsi kelayakan (*feasible*)
5. Fungsi obyektif

Dengan kata lain:

algoritma *greedy* melibatkan pencarian sebuah himpunan bagian, *S*, dari himpunan kandidat, *C* yang dalam hal ini, *S* harus memenuhi beberapa kriteria yang ditentukan, yaitu menyatakan suatu solusi dan *S* dioptimisasi oleh fungsi obyektif.

d. Kesimpulan dari Contoh kasus video :

- Pada setiap langkah, kita membuat pilihan **optimum lokal** (*local optimum*)
- dengan harapan bahwa langkah sisanya mengarah ke solusi **optimum global** (*global optimm*).
- Algoritma *greedy* adalah algoritma yang memecahkan masalah langkah per langkah; pada setiap langkah:

  1. mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan (prinsip "*take what you can get now!*")
  2. berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global.

3. Algoritma Prim

a. Pengertian / Definisi

Algoritme Prim adalah sebuah algoritme dalam teori graf untuk mencari pohon rentang minimum untuk sebuah graf berbobot yang saling terhubung. Ini berarti bahwa sebuah himpunan bagian dari edge yang membentuk suatu pohon yang mengandung node, di mana bobot keseluruhan dari semua edge dalam pohon diminimalisasikan. Bila graf tersebut tidak terhubung, maka graf itu hanya memiliki satu pohon rentang minimum untuk satu dari komponen yang terhubung

b. Penjelasan Singkat Alur

a. Tentukan titik awal sembarang pada graph
b. Masukan semua edge yang belum dijelajahi dari titik tersebut ke dalam acticeEdges selama edhe tersebut tidak menghasilkan looping
c. Pilih vertex dengan edge yang mempunyai weight terkecil di dalam ActiveEdges
d. Lakukan Kembali sampai semua vertex terjelajahi

c. Langkah Algoritma Prim

Algoritma Prim digunakan untuk mencari pohon pembangkit minimum dari graf terhubung berbobot dengan cara mengambil sisi/ ruas garis yang memiliki bobot terkecil dari graf, diamana ruas garis tersebut bersisian dengan pohon terentang yang telah dibuat dan yang tidak membentuk siklus. Untuk mencari pohon pembangkit minimum T dari graf G dengan Algoritma Prim dapat dilakukan dengan langkah-langkah sebagai berikut:

1. Ambilah satu simpul sembarang (misalnya v1 € G) dan masukkan simpul tersebut ke

dalam graf T yang merupakan graf kosong.

2. Tambahkanlah satu rusuk yang terhubung dengan v1 dengan bobot yang paling minimum

(misalnya e1) dan titik ujung lainnya ke T sehingga T terdiri dari sebuah rusuk e1 dan dua

buah simpul yang merupakan titik-titik ujung dari rusuk e1 (salah satu titik ujung harus memuat simpul v1).

3. Pada langkah berikutnya, pilihlah sebuah rusuk dalam E(G) yang bukan E(T) dengan syarat yang harus dipenuhi sebagai berikut:

a. Rusuk tersebut menghubungkan salah satu simpul V(T).

b. Rusuk tersebut mempunyai bobot minimal.

4. Ulangilah langkah tersebut (langkah 2-3) hingga diperoleh (n-1) rusuk dalam E(T) dengan n adalah banyaknya simpul dalam G. Dengan kata lain ulangi iterasi tersebut sebanyak (n-2)

d. Analisa Algoritma Prim

Algoritma Prim menggunakan strategi Greedy yaitu pada setiap langkah akan dipilih yang terbaik dan tidak mungkin mengulang langkah yang sebelumnya hingga akhir algoritma. Langkah pertama pada Algoritma Prim adalah memilih sebarang simpul pada graf G, kemudian mencari sisi pada himpunan E yang menyatakan sisi-sisi pada graf G dengan bobot terkecil kemudian dimasukan pada himpuan T. Setelah itu akan dilakukan perulangan/iterasi sebanyak n-2 untuk mencari sisi dengan bobot terkecil pada himpunan E yang bersisian dengan simpul yang telah dimasukan pada T. Hasil pencarian tersebut kemudian digabungkan atau ditambahkan pada himpunan T.

4. Algoritma Bellman-Ford

a. Pengertian / Definisi

Algoritma Bellman-Ford digunakan untuk mencari lintasan terpendek dari sumber certex (vertex awal) ke seluruh vertex yang memungkinkan. Dan juga Algoritma Bellman-Ford adalah salah satu algoritma yang digunakan untuk pencarian jalur. Contoh yang dibahas kali ini adalah mengenai pencarian jalur yang melalui semua titik dengan jarak terpendek.

Lalu algoritma ini juga menghitung jarak terpendek (dari satu sumber) pada sebuah digraf berbobot. Maksudnya dari satu sumber ialah bahwa ia menghitung semua jarak terpendek yang berawal dari satu titik node. Algoritme Dijkstra dapat lebih cepat mencari hal yang sama dengan syarat tidak ada sisi (edge) yang berbobot negatif. Maka Algoritme Bellman-Ford hanya digunakan jika ada sisi berbobot negatif.

Algoritma ini mirip dengan algoritma Djikstra tetapi Algoritma ini mampu menghitung path yang memiliki bobot negatif, namun waktu yang dibutuhkan oleh algoritma ini lebih lama dari pada algoritma Djikstra.

b. Cara Kerja Algoritma

Cara kerja algoritma ini dalam mencari jarak terpendek adalah dengan menghitung setiap kemungkinan node yang mengarah ke node tujuan tersebut. algoritma Bellman-Ford mengembalikan sebuah nilai Boolean yang mengindikasikan apakah terdapat siklus berbobot negatif yang dapat dilalui oleh simpul awal atau tidak. Jika terdapat siklus negatif algoritma akam mengindikasikan bahwa tidak terdapat solusi shortest path dan jika tidak, maka algoritma akan menghasilkan shortest path beserta bobotnya. Algoritma ini melakukan iterasi dalam setiap langkahnya sebanyak n-1, dimana adalah n jumlah node yang terdapat dalam graf. Dengandemikian kompleksitas algoritma ini cukup tinggi.

Kompleksitas waktu dari Algoritma Bellman-Ford ini dapat dinyatakan dengan notasi Big O(V.E), V adalah banyaknya sisi dan E adalah banyaknya titik.

5. Algoritma Djikstra

a. Pengertian / Definisi

Algoritme Dijkstra, (sesuai penemunya Edsger Dijkstra), adalah sebuah algoritma yang dipakai dalam memecahkan permasalahan jarak terpendek (shortest path problem) untuk sebuah graf berarah (directed graph). Permasalahan rute terpendek dari sebuah titik ke akhir titik lain adalah sebuah masalah klasik optimasi yang banyak digunakan untuk menguji sebuah algoritma yang diusulkan. Permasalahan rute terpendek dianggap cukup baik untuk mewakili masalah optimisasi, karena permasalahannya mudah dimengerti (hanya menjumlahkan seluruh edge yang dilalui) namun memiliki banyak pilihan solusi. Diskripsi matematis untuk grafik dapat diwakili G = {V. E}, yang berarti sebuah grafik (G) didefenisikan oleh satu set simpul (Vertex = V) dan koleksi Edge (E).

b. Cara Kerja Algoritma

Algoritma Dijkstra bekerja dengan membuat jalur ke satu simpul optimal pada setiap langkah. Jadi pada langkah ke n, setidaknya ada n node yang sudah kita tahu jalur terpendek. Langkah-langkah algoritma Dijkstra dapat dilakukan dengan langkah-langkah berikut:

1. Tentukan titik mana yang akan menjadi node awal, lalu beri bobot jarak pada node pertama ke node terdekat satu per satu, Dijkstra akan melakukan pengembangan pencarian dari satu titik ke titik lain dan ke titik selanjutnya tahap demi tahap.
2. Beri nilai bobot (jarak) untuk setiap titik ke titik lainnya, lalu set nilai 0 pada node awal dan nilai tak hingga terhadap node lain (belum terisi) 2.
3. Set semua node yang belum dilalui dan set node awal sebagai “Node keberangkatan”
4. Dari node keberangkatan, pertimbangkan node tetangga yang belum dilalui dan hitung jaraknya dari titik keberangkatan. Jika jarak ini lebih kecil dari jarak sebelumnya (yang telah terekam sebelumnya) hapus data lama, simpan ulang data jarak dengan jarak yang baru
5. Saat kita selesai mempertimbangkan setiap jarak terhadap node tetangga, tandai node yang telah dilalui sebagai “Node dilewati”. Node yang dilewati tidak akan

pernah di cek kembali, jarak yang disimpan adalah jarak terakhir dan yang paling minimal bobotnya.

6. Set “Node belum dilewati” dengan jarak terkecil (dari node keberangkatan) sebagai “Node Keberangkatan” selanjutnya dan ulangi langkah e.