PERTEMUAN 12:
GRAPH TERAPAN

A. TUJUAN PEMBELAJARAN

Pada bab ini akan dijelaskan mengenai lintasan dan sirkuit hamilton, Anda harus mampu:

1.1 Mengetahui Pohon dalam graph

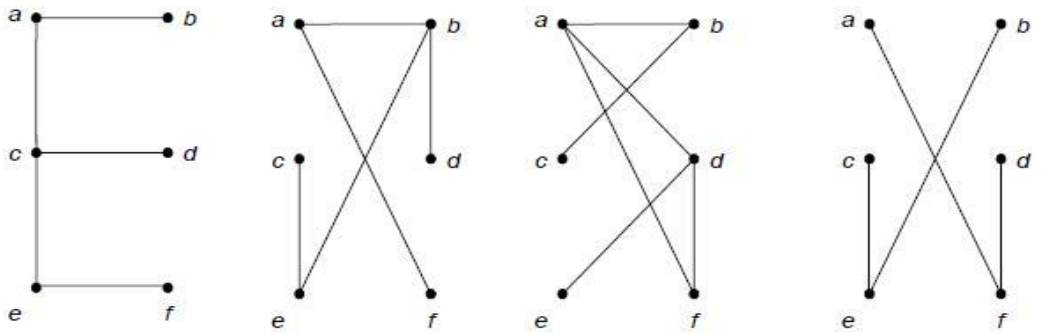
1.2 Mampu menyelesaikan persoalan dengan teori pohon

B. URAIAN MATERI

Tujuan Pembelajaran 1.1:
Mengetahui pohon dalam graph dan menyelesaikan permasalahan dengan teori tersebut

DEFINISI POHON

Pohon (*tree*) merupakan salah satu bentuk khusus dari struktur suatu graf. Misalkan A merupakan sebuah himpunan berhingga simpul (*vertex*) pada suatu graf ***G*** yang terhubung. Untuk setiap pasangan simpul di A dapat ditentukan suatu lintasan yang menghubungkan pasangan simpul tersebut. Suatu graf terhubung yang setiap pasangan simpulnya hanya dapat dihubungkan oleh suatu lintasan tertentu, maka graf tersebut dinamakan pohon (*tree*). Dengan kata lain, pohon (*tree*) merupakan graf tak-berarah yang terhubung dan tidak memiliki sirkuit.



G1 ***G2*** ***G3*** ***G4***

Gambar 3.1. ***G*** dan ***G*** adalah pohon, sedangkan ***G*** dan ***G*** bukan pohon
1 2 3 4

Hutan (*forest*) merupakan kumpulan pohon yang saling lepas. Dengan kata lain, hutan merupakan graf tidak terhubung yang tidak mengandung sirkuit. Setiap komponen di dalam graf terhubung tersebut adalah pohon. Pada gambar 2.1, G_4 merupakan salah satu contoh hutan, yaitu hutan yang terdiri dari dua pohon.

Sifat-sifat Pohon

Teorema 3.1.

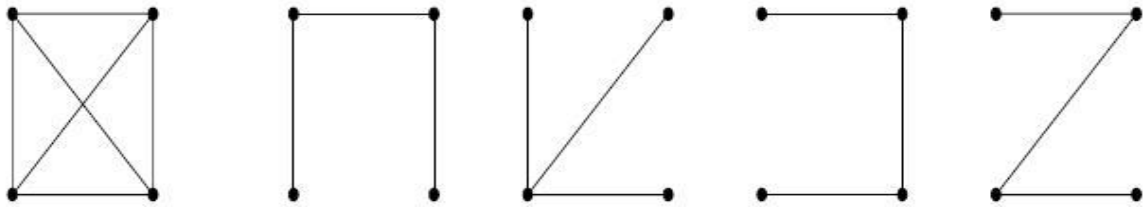
Misalkan $G = (V, E)$ adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka, semua pernyataan di bawah ini adalah ekivalen:

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki $m = n - 1$ buah sisi.
4. G tidak mengandung sirkuit dan memiliki $m = n - 1$ buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf hanya akan membuat satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

3.1 Pohon Merentang Minimum (*Minimum Spanning Tree*)

Spanning Tree dari suatu graf terhubung merupakan subgraf merentang yang berupa pohon. Pohon merentang diperoleh dengan cara menghilangkan sirkuit di dalam graf tersebut.

Contoh *spanning tree* dari suatu graf terhubung (**Munir, 2003**) :



G T_1 T_2 T_3 T_4

Gambar 3.2. Spanning Tree

Perhatikan graf di atas :

Terlihat bahwa T_1, T_2, T_3, T_4 merupakan s*panning tree* dari graf G. Perlu diperhatikan

bahwa setiap graf terhubung berbobot paling sedikit mempunyai satu buah *spanning tree*. Pohon rentang yang memiliki bobot minimum dinamakan pohon merentang minimum (*minimum spanning tree*). Dalam kehidupan nyata, salah satu contoh aplikasi s*panning tree* adalah menentukan rangkaian jalan dengan jarak total seminimum mungkin yang menghubungkan semua kota sehingga setiap kota tetap terhubung satu sama lain.

Dalam menentukan suatu *minimum spanning tree* dari suatu graf terhubung, kita dapat menentukannya dengan mengunakan dua cara yaitu algoritma Prim dan algoritma Kruskal. Algoritma Prim memiliki langkah-langkah sebagai berikut :

1. Pilih sisi dari graf G yang berbobot minimum, masukkan ke dalam T.

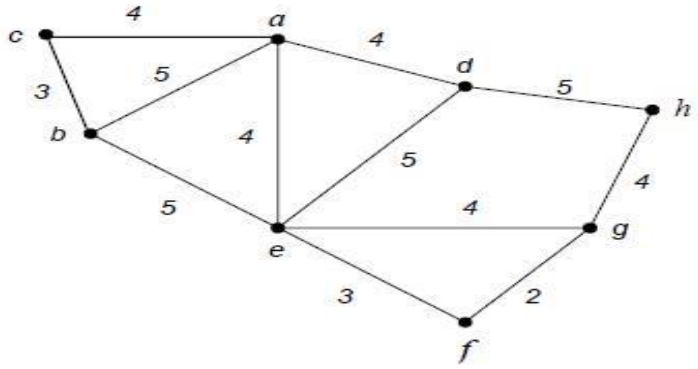
2. Pilih sisi (u, v) dalam G yang mempunyai bobot minimum dan bersisian dengan simpul di T, dengan syarat sisi tersebut tidak membentuk sirkuit di T. Masukkan (u, v) ke dalam T.

3. Ulangi langkah 2 sebanyak $n - 2$ kali.

Jumlah langkah seluruhnya dalam algoritma Prim adalah sebanyak jumlah sisi di dalam *spanning tree* dengan n buah simpul, yaitu ($n - 1$) buah.

Contoh 3.1:

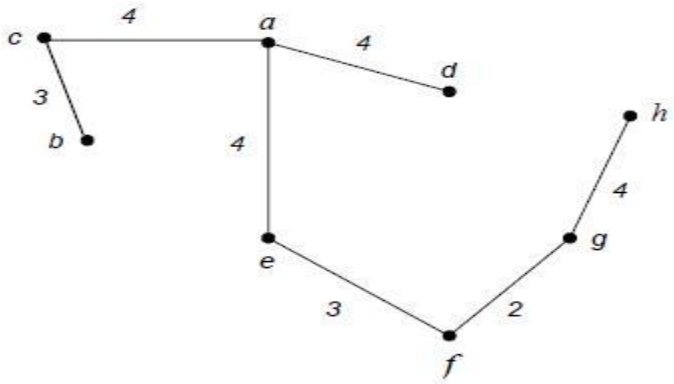
Tentukan *minimum spanning tree* dari graf dibawah ini :



Jawab :

- Pilih sisi *fg* sehingga kita mempunyai T ({*f*, *g*}, *fg*)
- Langkah selanjutnya dapat dipilih sisi *ef* karena sisi tersebut berbobot minimum yang bersisian dengan simpul *f* .
- Selanjutnya pilih sisi *ae* atau *gh* karena sisi tersebut berbobot minimum yang bersisian dengan simpul pada *T*, yaitu *e* dan *g*.

Jika proses ini dilanjutkan terus maka akan diperoleh *minimum spanning tree* seperti dibawah ini :

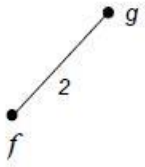


Terlihat bahwa *spanning tree* tersebut mempunyai total bobot : 2 + 3 + 4 + 4 + 4 + 4 + 3 = 24.

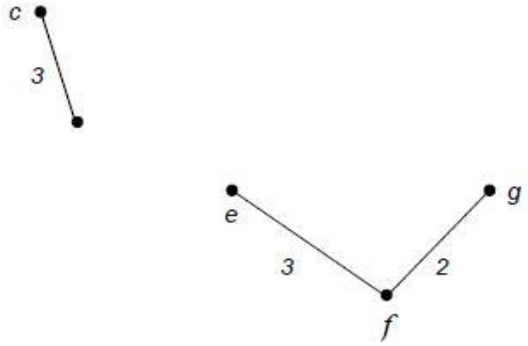
Langkah-langkah dalam algoritma Kruskal agak berbeda dengan algoritma Prim. Pada algoritma Kruskal, semua sisi dengan bobot yang minimal dimasukan kedalam T secara berurutan.

Langkah-langkah dalam menentukan *minimum spanning tree* dengan algoritma Kruskal adalah sebagai berikut :

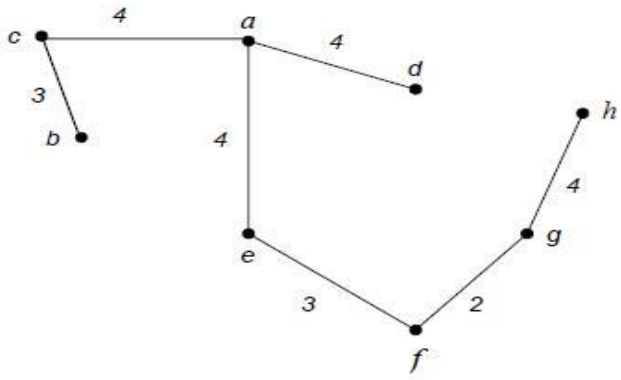
Langkah I : T berbentuk seperti pohon berikut



Langkah II : memasukan sisi-sisi yang berbobot 3 kedalam T sehingga berbentuk

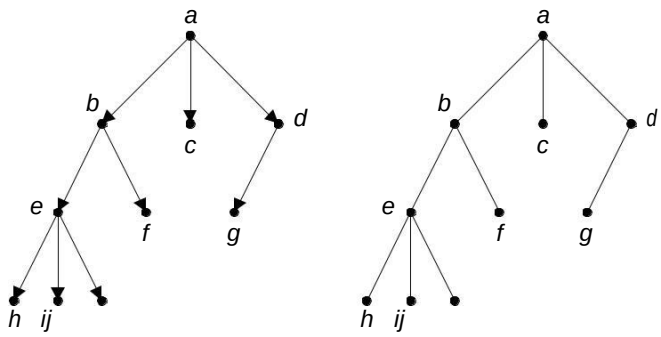


Langkah III : memasukan sisi-sisi yang berbobot 4 kedalam sehingga akhirnya diperoleh *minimum spanning tree* berikut :



3.2 Pohon Berakar

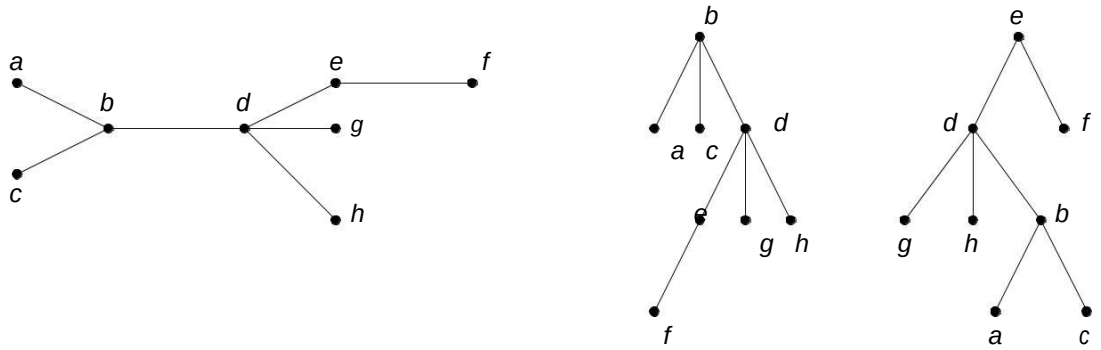
Pada suatu pohon, yang sisi-sisinya diberi arah sehingga menyerupai graf berarah, maka simpul yang terhubung dengan semua simpul pada pohon tersebut dinamakan **akar**. Suatu pohon yang satu buah simpulnya diperlakukan sebagai akar maka pohon tersebut dinamakan pohon berakar (*rooted tree*). Simpul yang berlaku sebagai akar mempunyai derajat masuk sama dengan nol. Sementara itu, simpul yang lain pada pohon itu memiliki derajat masuk sama dengan satu. Pada suatu pohon berakar, Simpul yang memiliki derajat keluar sama dengan nol dinamakan **daun**.



Gambar 3.3 : Pohon Berakar (**Munir, 2003**)

Pada pohon berakar diatas :

- *a* merupakan akar
- *c, f, g, h, i,* dan *j* merupakan daun



Gambar 3.4 : Pohon dan dua buah pohon berakar yang dihasilkan dari pemilihan dua simpul berbeda sebagaia akar

C. SOAL LATIHAN/TUGAS

DAFTAR PUSTAKA

Munir, Rinaldi. *Matematika Diskrit.* Bandung: Informatika, 2005.

Siang, Jong Jek. *Matematika Diskrit dan Aplikasinya pada Ilmu komputer.* Yogyakarta: Andi Offset, 2004.

Wibisono, Samuel. *Matematika Diskrit.* Yogyakarta: Graha Ilmu, 2008.