BAB II

TEORI GRAPH

Suatu Graph G merupakan himpunan pasangan berurutan (V,E) dimana :

$V = \{v_1, v_2, v_3, \ldots v_n\}$ adalah himpunan titik - titik v_i dan merupakan himpunan yang tidak kosong.

$E = \{e_1, e_2, e_3, \ldots e_n\}$ adalah himpunan garis - garis e_i yang menghubungkan dua titik $v_i \in V$, dimana himpunan E dapat berupa suatu himpunan kosong.

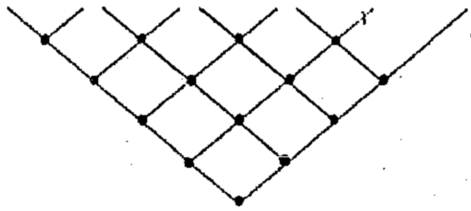
Dalam suatu graph, v_i berupa titik dan merupakan unsur atau elemen anggota dari V yang disebut simpul atau *vertex*, sedangkan e_i berupa garis yang merupakan unsur atau elemen anggota dari E yang disebut sisi atau *edge*, dimana himpunan E dapat berupa himpunan kosong.

Untuk memudahkan pengertian dari suatu graph, setiap vertex digambarkan sebagai suatu titik pada suatu bidang datar, sedangkan edge digambarkan sebagai garis yang menghubungkan dua titik pada graph tersebut.

Graph G adalah graph tidak berhingga (infinite graph) jika :

1. V tidak berhingga
2. E boleh berhingga

Contoh : 2.1.



Gambar 2.1. : Graph tidak berhingga

Graph G adalah graph berhingga (finite graph) jika :

1. V berhingga dan V $\neq \emptyset$
2. E berhingga

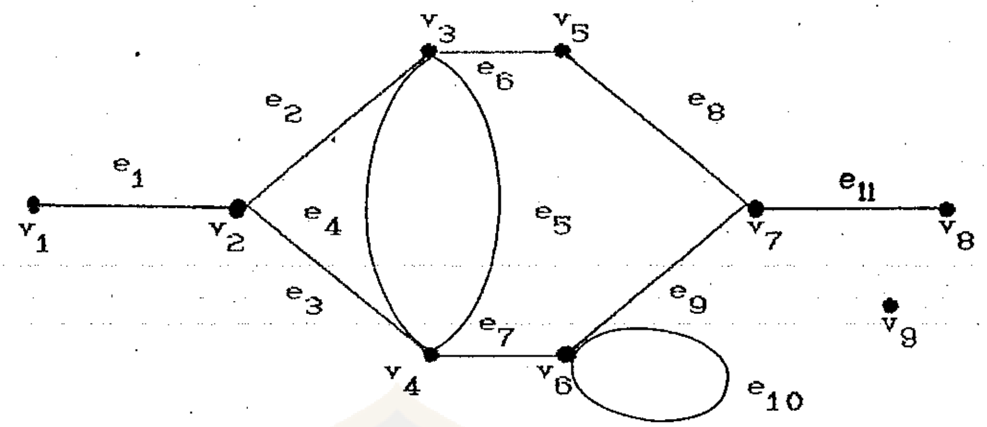
Telah disebut di atas bahwa penulisan ini akan membahas mengenai penggunaan komputer dalam membantu menyelesaikan beberapa algoritma dalam Teori Graph. Komputer akan dapat membantu menyelesaikan suatu permasalahan jika permasalahan tersebut terbatas. Oleh karena itu pembahasan dalam penulisan ini hanya akan dibatasi pada graph berhingga (finite graph).

Secara umum graph dapat dibedakan menurut jenisnya, yaitu **graph tidak berarah** atau yang biasa disebut **graph** saja dan **graph berarah**.

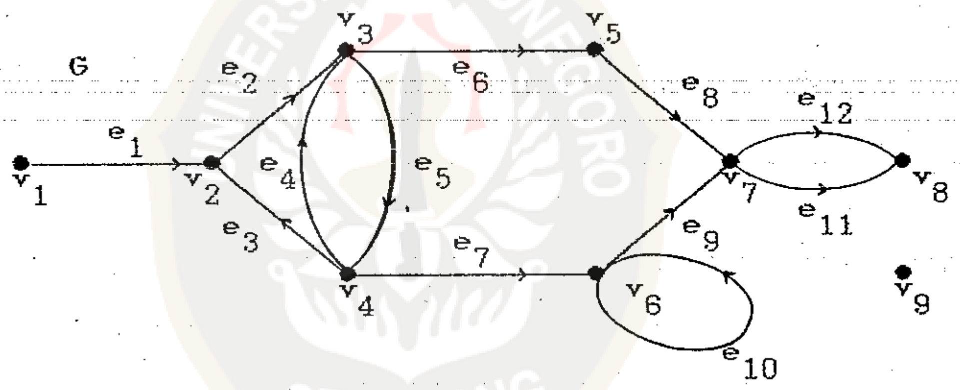
Definisi : 2.1.

Graph disebut sebagai **graph tidak berarah** apabila masing - masing edge pada graph tersebut tidak menentukan vertex awal (initial/source) dan vertex tujuan (terminal/target). Graph disebut sebagai **graph berarah** apabila masing - masing edge pada graph tersebut menentukan vertex awal (initial / source) dan vertex tujuan (terminal / target).

Contoh : 2.2.



Gambar : 2.2. Graph tidak berarah



Gambar : 2.3. Graph Berarah

2.1. Beberapa Pengertian Graph

Beberapa pengertian pada suatu Graph adalah :

- Loop

  Suatu edge dari Graph G yang berawal dan berakhir pada suatu vertex yang sama dinamakan Loop.

  Pada Gambar 2.2. dan Gambar 2.3. edge e_{10} adalah suatu loop.

- **Adjacent**

Pada suatu graph G, vertex v_1 dikatakan adjacent dengan v_2 apabila terdapat edge yang menghubungkan v_1 ke v_2. Jika tidak terdapat edge yang menghubungkan v_2 ke v_1, maka v_2 dikatakan tidak adjacent dengan v_1.

Jadi $v_1 \in V$ adjacent $v_2 \in V$ jika $\exists$ $e \in E$ yang menghubungkan v_1 dan v_2 .

Pada **Gambar 2.3.** v_1 adjacent dengan v_2, tetapi v_2 tidak adjacent dengan v_1, v_3 adjacent v_4, dan v_4 adjacent v_3.

- **Parallel edge atau garis paralel**

Pada suatu graph tak berarah G, beberapa edge yang bersama - sama menghubungkan dua vertex yang sama disebut parallel edge.

Pada suatu graph berarah G, beberapa edge yang memiliki initial dan terminal yang sama disebut parallel edge.

Pada **Gambar 2.2.** edge e_4 dan e_5 adalah parallel edge yang bersama-sama menghubungkan v_3 dan v_4.

Pada **Gambar 2.3.** edge e_{11} dan e_{12} adalah parallel edge, tetapi edge e_4 dan e_5 bukan merupakan parallel edge.

- **Derajat atau degree**

Pada suatu graph tak berarah G banyaknya edge yang bertemu pada suatu vertex menunjukkan derajat dari vertex tersebut ($d(v_i)$).

Pada suatu graph berarah G, derajat dibedakan menjadi dua, yaitu banyaknya edge yang menuju ke suatu vertex yang dinamakan *in degree* $(d.in(v_i))$ dan banyaknya edge yang meninggalkan suatu vertex yang dinamakan *out degree* $(d.out(v_i))$.

Theorema : 2.1.

Jumlah semua degree dalam suatu graph = dua kali banyaknya edge.

Bukti :

Setiap edge pada suatu graph G disangga oleh vertex pada setiap ujungnya, sehingga setiap edge menambah 1 derajat pada masing - masing vertex yang menyangganya. Maka jumlah derajat pada suatu graph G adalah 2 kali banyaknya edge dalam graph tersebut.

$(\sum d(v_i) = 2e)$.

Pada Gambar 2.3., jumlah vertex = 9,

jumlah edge = 12.

$$\begin{array}{lr} (d.in(v_1)) & = 0 \\ (d.out(v_1)) & = 1 \\ \hline d(v_1) & = 1 \end{array} \qquad \begin{array}{lr} (d.in(v_2)) & = 2 \\ (d.out(v_2)) & = 1 \\ \hline d(v_2) & = 3 \end{array}$$

$$\begin{array}{lr} (d.in(v_3)) & = 2 \\ (d.out(v_3)) & = 2 \\ \hline d(v_3) & = 4 \end{array} \qquad \begin{array}{lr} (d.in(v_4)) & = 1 \\ (d.out(v_4)) & = 3 \\ \hline d(v_4) & = 4 \end{array}$$

$$\begin{array}{rl} (d.in(v_5)) & = 1 \\ (d.out(v_5)) & = 1 \\ \hline d(v_5) & = 2 \end{array} \qquad \begin{array}{rl} (d.in(v_6)) & = 2 \\ (d.out(v_6)) & = 2 \\ \hline d(v_6) & = 4 \end{array}$$

$$\begin{array}{rl} (d.in(v_7)) & = 2 \\ (d.out(v_7)) & = 2 \\ \hline d(v_7) & = 4 \end{array} \qquad \begin{array}{rl} (d.in(v_8)) & = 2 \\ (d.out(v_8)) & = 0 \\ \hline d(v_8) & = 2 \end{array}$$

$$\begin{array}{rl} (d.in(v_9)) & = 0 \\ (d.out(v_9)) & = 0 \\ \hline d(v\ ) & = 0 \end{array}$$

Total degree = $\sum d(v_i)$ = 24 = 2 edge.

Sesuai dengan Theorema 3.1. bahwa dalam suatu graph, jumlah seluruh degree = dua kali banyaknya edge.

- **Isolated Vertex atau Titik Terasing**

  Suatu vertex yang tidak memiliki derajat atau derajat vertex tersebut = nol disebut isolated vertex.

  Pada Gambar 2.2. dan Gambar 2.3. vertex v_9 adalah isolated vertex.

- **End Vertex atau Titik Akhir dan Pendan**

  Pada suatu graph tak berarah G suatu vertex yang hanya memiliki derajat satu dinamakan end vertex.

  Pada suatu graph berarah G suatu vertex yang hanya memiliki derajat satu dinamakan pendan.

  Atau $\sum d(v_i) = d.in(v_i) + d.out(v_i) = 1$.

Pada Gambar 2.2. vertex v_1 dan v_8 adalah end vertex.

Pada Gambar 2.3. vertex v_1 adalah suatu pendan.

- **Graph Nol atau Null Graph**

  Suatu Graph G yang setiap vertex di dalamnya berderajat nol, atau dengan kata lain suatu graph yang tidak memiliki edge.

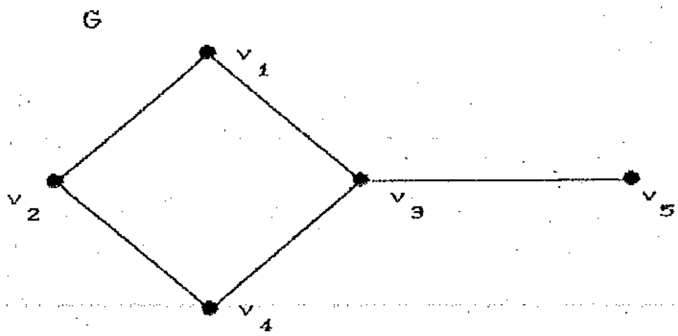
Contoh 2.3 :

Gambar 2.4. Graph nol atau null graph

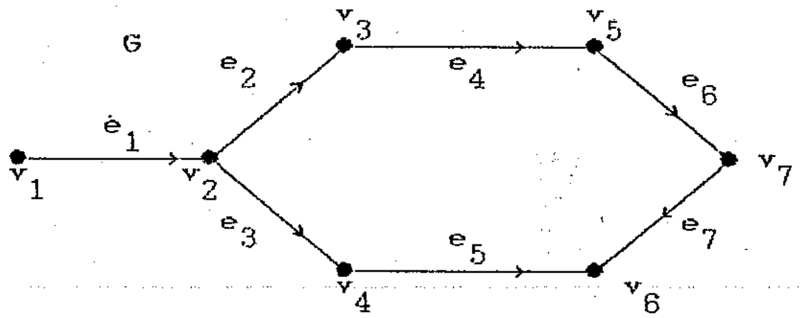
- **Graph Sederhana**

  Graph yang tidak memiliki loop dan parallel edge dinamakan graph sederhana.

Contoh : 2.4.



Gambar 2.5. : Graph tak berarah sederhana



Gambar 2.6. Graph berarah sederhana

- Graph Berarah Simetris (Symetric Digraph)

  Suatu graph berarah G dinamakan graph berarah simetris jika graph berarah tersebut mempunyai edge (v_i, v_j) (yaitu edge dari vertex v_i ke v_j), maka juga memiliki edge (v_j, v_i).

Contoh 2.5. :

Gambar 2.7. Graph Berarah Simetris

- Graph Berarah Asimetris

  Suatu graph berarah G dinamakan graph berarah asimetris jika pada graph berarah tersebut setiap pasang vertexnya dihubungkan paling banyak oleh satu edge.

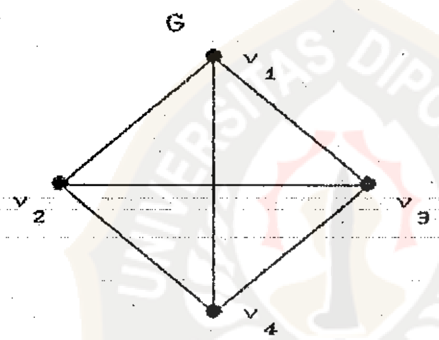
  Gambar 2.6. adalah graph berarah asimetris.

- Graph Komplit dan Graph Berarah Komplit Simetris

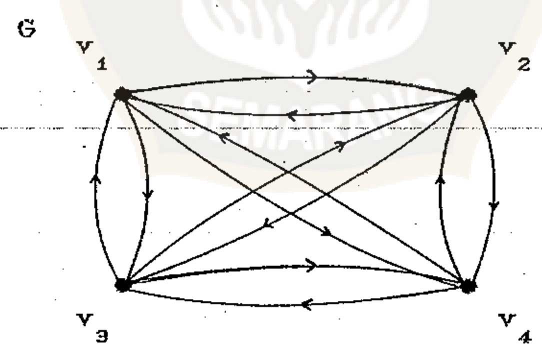
  Suatu graph tak berarah G yang setiap vertex pada graph tersebut adjacent dengan setiap vertex lainnya dinamakan graph komplit.

  Suatu graph berarah G yang setiap vertex pada graph tersebut saling adjacent dengan setiap vertex lainnya dinamakan graph berarah komplit simetris.

Contoh : 2.6.



Gambar 2.8. : Graph Komplit

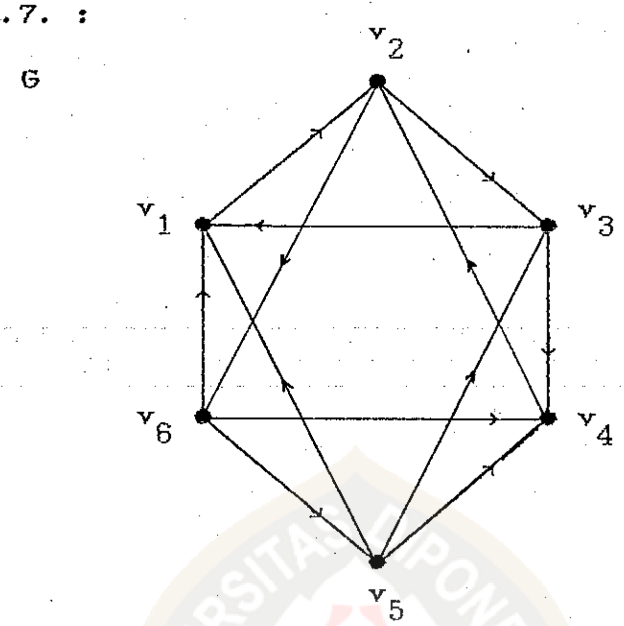


Gambar 2.9. Graph Berarah Komplit Simetris

- Graph Berarah Komplit Asimetris

  Suatu graph berarah G yang setiap vertexnya dihubungkan dengan vertex lain, dan paling banyak dihubungkan oleh satu edge dinamakan graph berarah komplit asimetris.

Contoh 2.7. :

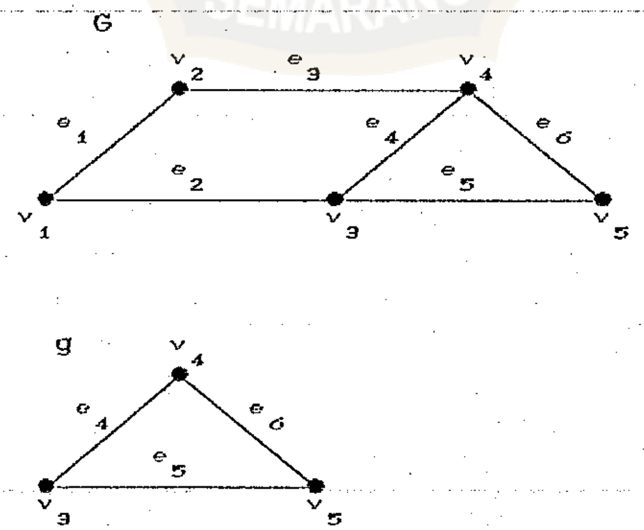


Gambar 2.10 Graph Berarah Komplit Asimetris

- Sub Graph

Graph tak berarah g adalah sub graph dari graph tak berarah G apabila setiap vertex dan setiap edge dari graph g berada dalam graph G.

Contoh : 2.8.



Gambar 2.11 : graph g adalah sub graph dari graph G

- Lintasan (path)

  Rute perjalanan dari suatu vertex ke vertex yang lain dalam suatu graph G yang ditunjukkan oleh barisan vertex dan edge, yang diawali dan diakhiri oleh vertex.

  Pada Gambar 2.2. salah satu lintasan dari v_1 hingga v_8 adalah :

  $v_1 - e_1 - v_2 - e_2 - v_3 - e_6 - v_5 - e_8 - v_7 - e_{11} - v_8$

- Lintasan Searah (Directed Path)

  Lintasan yang ditunjukkan sesuai dengan arah dari edgenya. Pada Gambar 2.3., salah satu lintasan searah dari v_1 ke v_4 adalah :

  $v_1 - e_1 - v_2 - e_2 - v_3 - e_6 - v_5 - e_8 - v_7 - e_{11} - v_8$

- Semi Lintasan (Semi Path)

  Lintasan yang bukan merupakan lintasan searah. Pada Gambar 2.3. salah satu semi lintasan ditunjukkan oleh :

  v_1 , e_1 , v_2 , e_2 , v_3 , e_4 , v_4.

- Sirkuit

  Suatu lintasan yang memiliki vertex awal dan vertex akhir sama atau biasa disebut lintasan tertutup.

  Salah satu sirkuit pada Gambar 2.2. adalah :

  $v_2 - e_2 - v_3 - e_6 - v_5 - e_8 - v_7 - e_9 - v_6 - e_7 - v_4 - e_3 - v_2$

- Sirkuit Searah (Directed Circuit)

  Sirkuit yang ditunjukkan sesuai dengan arah dari edgenya.

Pada Gambar 2.3. salah satu sirkuit searah ditunjukkan oleh :

v_2 , e_2 , v_3 , e_5 , v_4 , e_3 , v_2

- Semi Sirkuit (Semi Circuit)

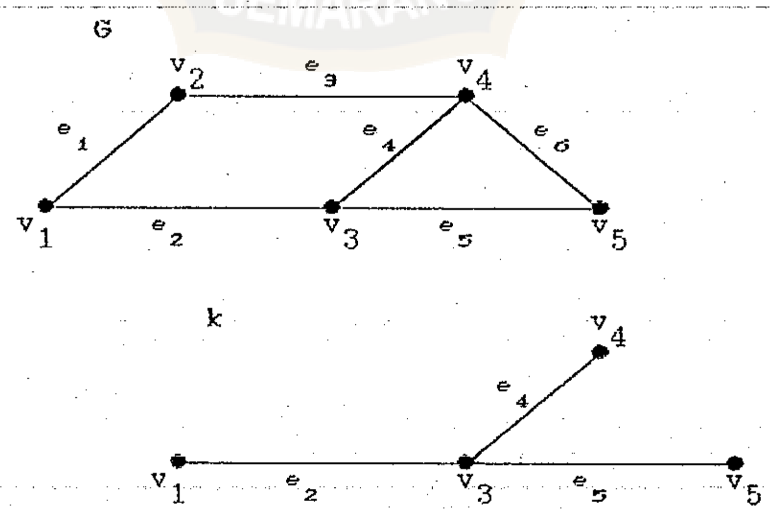
Sirkuit yang bukan merupakan sirkuit searah. Pada Gambar 2.3. salah satu semi sirkuit ditunjukkan oleh ;

$v_2 - e_2 - v_3 - e_6 - v_5 - e_8 - v_7 - e_9 - v_6 - e_7 - v_4 - e_3 - v_2$

- Komponen (component)

Graph tak berarah k disebut komponen dari graph tak berarah G, jika diambil vertex v_i yang berada dalam G, maka v_i dan semua vertex dalam G yang terhubung dengan v_i, dan semua edge yang menghubungkan, membentuk suatu sub graph yang disebut komponen.

Contoh : 2.9.



Gambar 2.12. : k adalah komponen dari graph G

- Komponen Kuat

  Suatu sub graph yang terhubung kuat maksimal dari suatu graph berarah dinamakan komponen kuat.

- Komponen Lemah

  Suatu sub graph yang terhubung lemah maksimal dari suatu graph berarah dinamakan komponen lemah.

- Connected Graph

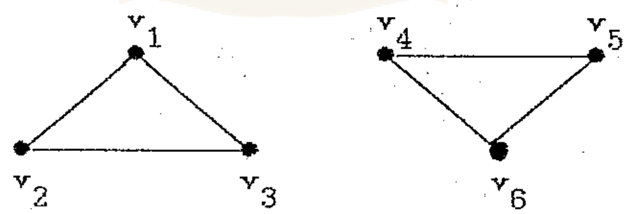
  Suatu graph tak berarah G disebut connected graph apabila setiap pasang vertex dalam graph tersebut terdapat paling sedikit satu lintasan.

- Disconnected Graph

  Suatu graph G disebut disconnected graph apabila sedikitnya terdapat sepasang vertex yang tidak dihubungkan oleh suatu lintasan.

  Contoh pada Gambar 2.13. tidak ada lintasan yang dapat dibuat dari v_1 hingga v_4.

Contoh : 2.10.



Gambar 2.13. : Disconnected Graph

Theorema : 2.2.

Graph G adalah graph tidak terhubung jika dan hanya jika himpunan vertex - vertex V dapat dipisahkan menjadi 2 himpunan vertex yaitu V_1 dan V_2 yang masing - masing tidak kosong, dan tidak ada lintasan yang menghubungkan vertex dalam V_1 dengan vertex dalam V_2.

Graph G tidak terhubung $\Longleftrightarrow$ tidak terdapat lintasan antara vertex a dan b, untuk $a \in V_1$ $b \in V_2$ dan $V_1 \cup V_2 = V$.

Bukti :

$\Longrightarrow$ G(V,E) tidak terhubung. Vertex a berada dalam G. Setiap vertex yang terhubung dengan a merupakan elemen V_1. Karena G tidak terhubung, maka terdapat V_2 dimana $V_2 = V - V_1$.
Maka terdapat vertex $b \in V_2$.
Maka jika graph G tidak terhubung, tidak terdapat lintasan yang menghubungkan vertex a dan b, dimana $a \in V_1$, $b \in V_2$ dan $V_1 \cup V_2 = V$.

$\Longleftarrow$ Tidak terdapat lintasan yang menghubungkan vertex a dan b, untuk $a \in V_1$, $b \in V_2$ dan $V_1 \cup V_2 = V$.
$a \in V_1$ dan $b \in V_2$, maka $V_1 \neq 0$ dan $V_2 \neq 0$. Tidak terdapat lintasan yang menghubungkan a dan b, maka tidak ada $e \in E$ yang menghubungkan vertex - vertex di V_1 dan V_2. Jadi G(V,E) tidak terhubung.

Theorema : 2.3.

Jika dalam G terdapat tepat 2 vertex yang berderajat ganjil, maka terdapat lintasan di antara 2 vertex tersebut.

Bukti :

Misal semua vertex dalam graph G berderajat genap kecuali v_1 dan v_2 yang berderajat ganjil. Karena banyaknya vertex berderajat ganjil adalah genap, maka v_1 dan v_2 berada dalam suatu komponen sehingga ada lintasan yang menghubungkan v_1 dan v_2.

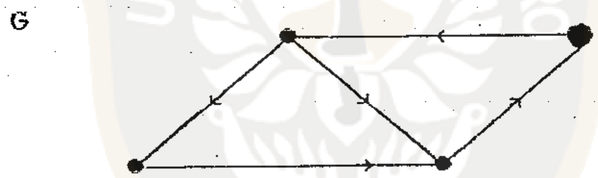
- Terhubung Kuat (Strongly Connected)

  Suatu graph berarah G disebut terhubung kuat jika sedikitnya terdapat satu lintasan searah dari setiap vertex ke setiap vertex yang lain. Graph berarah pada Gambar 2.14. adalah suatu graph terhubung kuat.

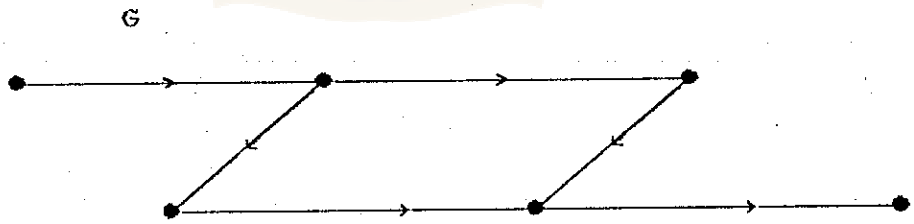
- Terhubung Lemah (Weakly Connected)

  Suatu graph berarah G disebut terhubung lemah jika graph tersebut terhubung, tetapi tidak terhubung kuat. Graph berarah pada Gambar 2.15. adalah suatu graph terhubung lemah.

Contoh 2.11. :



Gambar 2.14. Graph Terhubung Kuat



Gambar 2.15. Graph Terhubung Lemah

2.2. Tree

Tree adalah graph terhubung yang tidak memiliki sirkuit.

Theorema : 2.4.

Dalam tree T hanya terdapat satu lintasan pada setiap pasang vertex - vertexnya.

Bukti :

Tree adalah graph terhubung, maka terdapat lintasan yang menghubungkan setiap vertex dengan vertex - vertex yang lain. Tree tidak memiliki sirkuit, maka lintasan yang menghubungkan vertex - vertexnya hanya terdapat satu lintasan (tunggal).

Theorema : 2.5.

Jika dalam graph G hanya terdapat satu lintasan pada setiap pasang vertex - vertexnya, maka G adalah suatu tree.

Bukti :

Terdapat lintasan dalam graph G, sehingga G adalah graph terhubung. Hanya terdapat satu lintasan pada setiap pasang vertex - vertex dalam graph G, maka G tidak memiliki sirkuit. Jadi G adalah suatu tree.

Theorema : 2.6.

Suatu tree T dengan n vertex mempunyai n - 1 edge.

Bukti :

Karena T adalah suatu tree, maka T tidak memiliki sirkuit, jadi T adalah graph terhubung minimal. Jadi bila tree T memiliki n vertex, maka T memiliki n - 1 edge.

Theorema : 2.7.

Graph G terhubung dengan n vertex dan (n-1) edge adalah tree.

Bukti :

Graph G terhubung dengan n vertex dan (n-1) edge berarti G terhubung minimal. Jadi G tidak memiliki sirkuit. Sehingga G adalah tree.

Theorema : 2.8.

Graph G adalah suatu tree <==> G terhubung minimal.

Bukti :

==> G adalah tree, maka ada lintasan yang menghubungkan setiap pasang vertex - vertexnya dan G tidak memiliki sirkuit.

Andaikan G terhubung tidak minimal, sehingga ada e_i sedemikian bila e_i dihilangkan maka G terhubung minimal. Ini berarti bahwa e_i berada dalam suatu sirkuit.

Kontradiksi dengan pernyataan bahwa G adalah suatu tree, maka G tidak memiliki sirkuit. Pengandaian salah, maka G adalah terhubung minimal.

<== G terhubung minimal, maka G adalah tree.

G terhubung minimal, maka G tidak memiliki sirkuit. Jadi G adalah suatu tree.

Theorema : 2.9.

Graph G dengan n vertex, (n-1) edge dan tidak mempunyai sirkuit adalah terhubung.

Bukti :

Misal G memiliki n vertex, (n-1) edge dan tidak mempunyai sirkuit dan tidak terhubung. Berarti G terdiri atas lebih dari 1 komponen. Misal G terdiri atas 2 komponen. Diberikan edge e_{ij} yang menghubungkan v_i di G_1 dan v_j di G_2. Karena G_1 dan G_2 tidak memiliki sirkuit, maka $G' = G \cup e_{ij}$ juga tidak memiliki sirkuit. Berarti G' merupakan suatu tree dengan n vertex dan $(n-1) + 1 = n$ edge. Kontradiksi dengan suatu tree memiliki n vertex, $(n-1)$ edge dan tidak mempunyai sirkuit. Jadi pengandaian salah, sehingga G adalah terhubung.

Dari 6 theorema mengenai tree dapat disimpulkan bahwa suatu graph G dengan n vertex disebut **tree** jika :

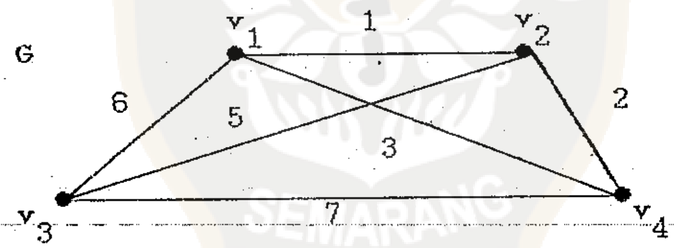
1. G terhubung dan tidak mempunyai sirkuit.
2. G terhubung dan hanya memiliki $(n-1)$ edge.
3. G tidak mempunyai sirkuit dan hanya memiliki $(n-1)$ edge.
4. Setiap pasang vertex dalam G hanya dihubungkan oleh satu lintasan .
5. G adalah terhubung minimal.

2.2.1. Spanning Tree

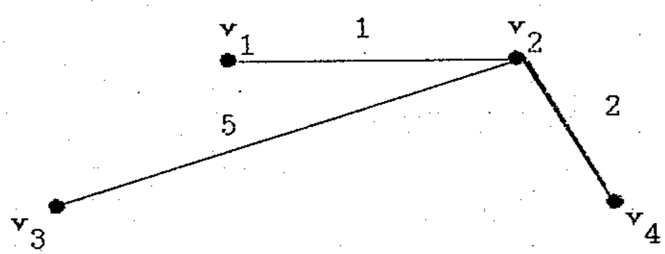
Spanning tree dari suatu graph G adalah suatu tree dimana setiap vertex dari tree adalah seluruh vertex dari G dan edge dari tree itu adalah sebagian edge dari graph G. Salah satu Algoritma Spanning Tree Minimal yang juga disebut sebagai Algoritma Kruskal adalah sebagai berikut :

1. Tentukan edge e_1 yaitu edge yang bermuatan terkecil (jika ada lebih dari satu, pilih salah satu).
2. Tentukan edge e_2 yang bermuatan terkecil antara $G-e_1$ (edge yang belum dipilih).
3. Proses diulang sampai didapatkan Spanning Tree, dengan syarat yang harus selalu dipenuhi bahwa edge yang dipilih tidak boleh menghasilkan sirkuit, jika ada lewati edge tersebut.
4. .Spanning Tree yang didapat bermuatan minimal.

Contoh : 2.12.



Gambar 2.16.



b

Gambar 2.17. :

Gambar 2.17. adalah Spanning Tree Minimal dari graph tak berarah G pada Gambar 2.16.