BAB II

MATERI PENUNJANG

2.1 Graph Tak Berarah

2.1.1 Pengertian

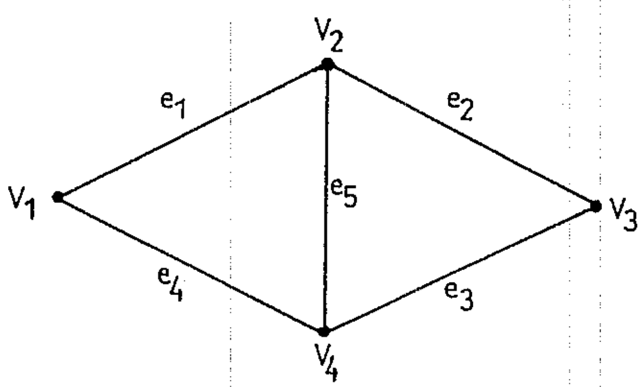
Definisi 2.1.1

Suatu graph G dinotasikan G=(V,E) adalah himpunan titik (V) dengan $V=\{v_1,v_2,v_3,\ldots,v_n\}$ yang berhingga dan tidak kosong, dan himpunan garis (E) dengan $E=\{e_1,e_2,e_3,\ldots,e_m\}$.

Definisi 2.1.2

Graph tak berarah adalah graph yang semua garisnya tidak mempunyai arah .

Contoh :

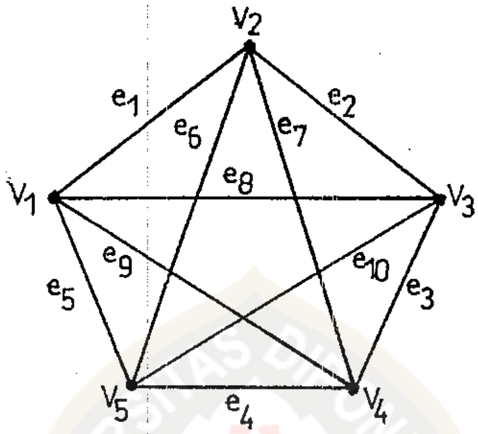


Gambar 2.1.1
Suatu Graph tak berarah dengan $V=\{v_1,v_2,v_3,v_4\}$ dan $E=\{e_1,e_2,e_3,e_4,e_5\}$

Definisi 2.1.3

Suatu graph lengkap (Complete Graph) adalah graph yang untuk setiap dua titik dihubungkan oleh satu garis.

Contoh :

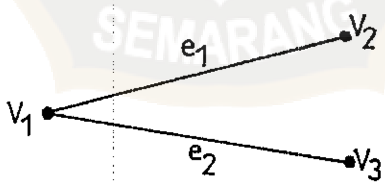


Gambar 2.1.2
Graph lengkap dengan lima titik

Definisi 2.1.4

Titik v_i dan v_j disebut titik akhir dari e_R jika e_R menghubungkan titik v_i dan v_j.

Contoh :



Gambar 2.1.3
Titik v_1 dan v_2 adalah titik akhir dari e_1, titik v_1 dan v_3 adalah titik akhir dari e_2

Definisi 2.1.5

Suatu garis e_i dikatakan incident dengan titik v_j jika v_j titik akhir dari e_i.

Contoh :

Pada gambar 2.1.1 garis e_1 dan e_4 incident dengan titik v_1.

Definisi 2.1.6

Dua titik dikatakan adjacent jika dihubungkan oleh sebuah garis.

Contoh :

Pada gambar 2.1.1, titik v_1 adjacent dengan v_2.

Definisi 2.1.7

Dua garis dikatakan adjacent jika keduanya incident pada titik yang sama.

Contoh :

Pada gambar 2.1.1, e_1 dan e_4 adjacent.

Definisi 2.1.8

Derajat (degree) dari titik v_i dinotasikan $d(v_i)$ adalah banyaknya garis yang incident dengan titik v_i.

Contoh :

Pada gambar 2.1.1, $d(v_1) = d(v_3) = 2$

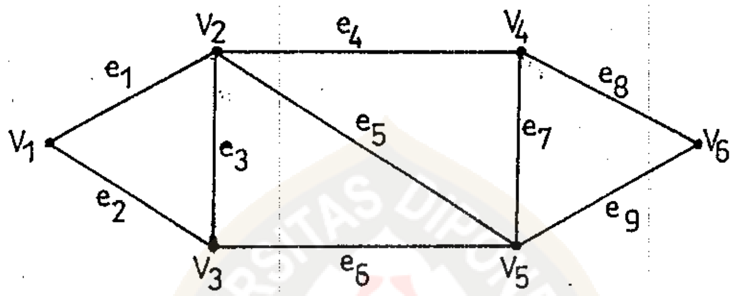
$$d(v_2) = d(v_4) = 3$$

Diberikan suatu graph G seperti pada gambar 2.1.4, dengan titik-titiknya menunjukkan kota dan garis-garisnya menunjukkan jalan. Dari kota v_1 ke kota v_6 terdapat barisan garis-garis yang merupakan rute dari v_1 ke v_6.

Suatu titik berkorespondensi ke asal dinamakan titik awal (Titik Initial) dan titik yang berkorespondensi ke tujuan dinamakan titik akhir (Titik Final).

Contoh :

Pada gambar 2.1.4 dengan titik awal v_1 dan titik akhir v_6 akan terdapat beberapa garis misalnya : (e_1,e_4,e_8), (e_1,e_3,e_6,e_9), (e_2,e_6,e_9), (e_2,e_3,e_5,e_9).



Gambar 2.1.4
Suatu graph G dengan titik awal v_1 dan titik akhir v_6

Yang harus dicatat bahwa setiap garis dalam suatu barisan yang dibicarakan mempunyai satu titik awal yang bersamaan dengan garis sebelumnya dan satu titik akhir dengan garis sesudahnya.

Contoh :

Dalam barisan (e_1,e_4,e_8) pada contoh di atas, titik v_2 adalah titik awal dari e_4 juga merupakan titik akhir dari garis sebelumnya yaitu e_1, dan titik v_4 adalah titik akhir dari e_4 juga merupakan titik awal dari garis sesudahnya yaitu e_8.

Jika setiap garis yang demikian dalam suatu barisan muncul hanya sekali, barisan ini dinamakan suatu train garis. Sehingga didapat definisi train garis sebagai berikut.

Definisi 2.1.9

Train garis adalah barisan garis-garis dengan sifat sebagai berikut :

(i) Untuk garis e_u selain pertama dan terakhir garis-garis dalam barisan, satu titik awal dari e_u adalah titik akhir dari garis sebelumnya dan titik akhir dari e_u adalah titik awal dari garis sesudahnya.

(ii) Satu titik akhir dari garis sebelumnya adalah titik awal dari garis e_u dan titik awal dari garis pertama adalah titik initial.

(iii) Satu titik awal dari garis terakhir adalah titik akhir dari garis e_u dan titik akhir dari garis terakhir adalah titik final.

(iv) Setiap garis muncul tepat satu kali.

Definisi 2.1.10

Suatu train garis disebut train garis terbuka jika titik initial dan titik finalnya berbeda, sebaliknya jika titik initial dan titik finalnya sama disebut train garis tertutup.

Contoh :

Pada gambar 2.1.4, barisan $(e_1,e_5,e_7,e_8,e_9,e_6)$ adalah train garis terbuka dengan titik awalnya v_1 dan titik akhirnya v_3. Barisan (e_1,e_3,e_2) adalah train garis tertutup.

Definisi 2.1.11

Path antara titik i dan j adalah himpunan semua garis dalam train garis terbuka yang memenuhi sifat sebagai berikut :

(i) Titik awal dan titik akhir adalah i dan j.

(ii) Setiap titik selain i dan j derajatnya 2 dan titik i dan j derajatnya 1.

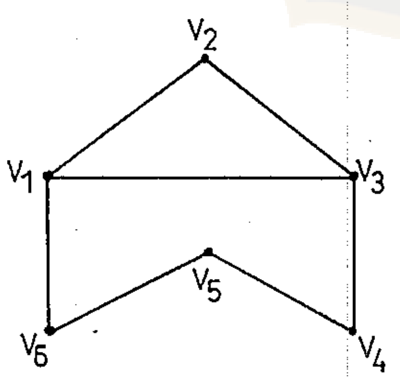
Contoh :

Himpunan semua garis dalam train garis (e_1, e_5, e_9) pada gambar 2.1.4 merupakan path antara titik v_1 dan v_6.

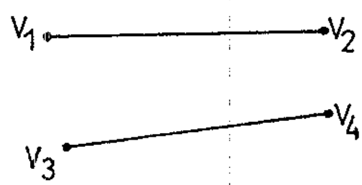
Definisi 2.1.12

Graph terhubung adalah jika setiap pasang titik-titiknya dihubungkan oleh path. Dan sebaliknya disebut graph tak terhubung.

Contoh :



Gambar 2.1.5a
Graph terhubung



Gambar 2.1.5b
Graph tak terhubung

Definisi 2.1.13

Sirkuit adalah graph terhubung yang setiap titiknya berderajat 2.

Contoh :

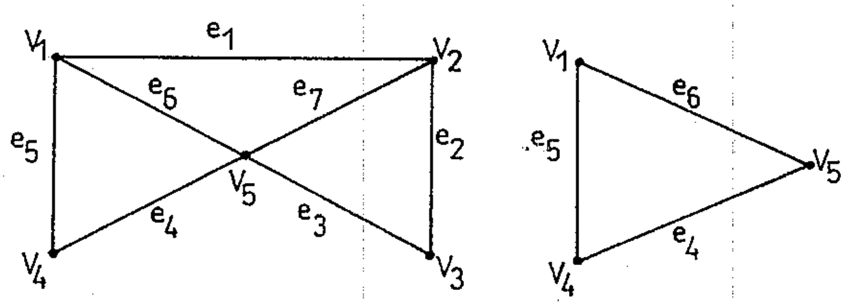
Himpunan garis dalam train garis tertutup (e_1,e_3,e_2) pada gambar 2.1.4 merupakan sirkuit tetapi himpunan garis dalam train garis tertutup $(e_1,e_5,e_7,e_8,e_9,e_6,e_2)$ pada gambar yang sama bukan merupakan sirkuit.

Definisi 2.1.14

Suatu graph g dikatakan subgraph dari G jika seluruh titik dan garisnya berada dalam G.

Sebuah subgraph bisa dikatakan sebagai graph yang termuat atau merupakan bagian dari graph yang lain. Simbol teori himpunan $g \subset G$, digunakan dengan kalimat g adalah sebuah subgraph dari G.

Contoh :



Gambar 2.1.6a Gambar 2.1.6b

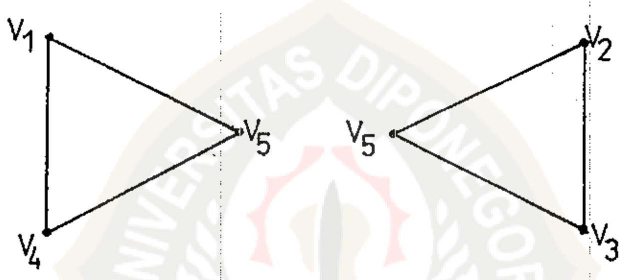
Gambar 2.1.6b merupakan salah satu subgraph dari graph pada gambar 2.1.6a

Definisi 2.1.15

Dua subgraph g_1 dan g_2 dari graph G disebut garis terpisah jika g_1 dan g_2 tidak mempunyai garis bersama-sama.

Contoh:

g_1 dan g_2 pada gambar 2.1.7 merupakan garis terpisah dari graph G pada gambar 2.1.6a.

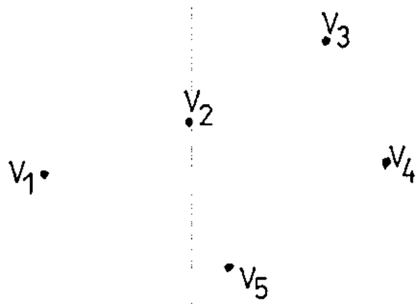


Gambar 2.1.7
Garis terpisah dari graph G pada gambar 2.1.6a

Definisi 2.1.16

Suatu graph G=(V,E) dikatakan null graph jika E kosong atau graph tersebut tidak mempunyai garis.

Contoh :



Gambar 2.1.8
Suatu null graph dengan $V=\{v_1, v_2, v_3, v_4, v_5\}$

2.1.2 Beberapa Operasi dalam Graph

Definisi 2.1.17

Union dari dua graph misal $G_1=(V_1,E_1)$ dan $G_2=(V_2,E_2)$ adalah graph lain misal G_3 yang ditulis $G_3=G_1 \cup G_2$, dengan himpunan titiknya $V_3=V_1 \cup V_2$ dan himpunan garisnya $E_3=E_1 \cup E_2$.

Definisi 2.1.18

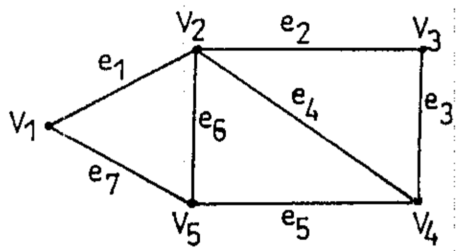
Irisan dari dua graph G_1 dan G_2 adalah graph lain G_4 yang ditulis $G_4=G_1 \cap G_2$ dengan himpunan titiknya $V_4=V_1 \cap V_2$ dan himpunan garisnya $E_4=E_1 \cap E_2$.

Definisi 2.1.19

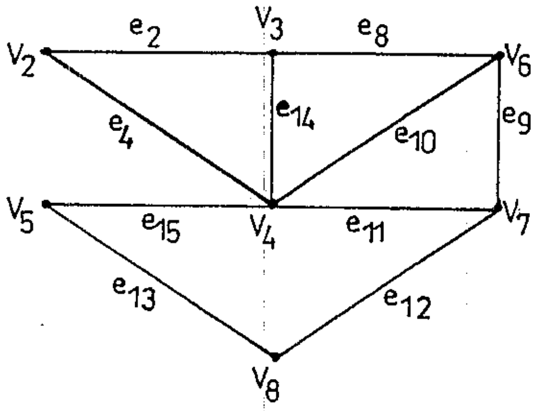
Operasi Jumlahan Cincin (Ring Sum) dengan notasi $\oplus$ antara dua graph G_1 dan G_2 adalah graph lain G_5 yang ditulis $G_5=G_1 \oplus G_2$ yang memenuhi $V_5=V_1 \cup V_2$ dan $E_5=(E_1 \cup E_2) - (E_1 \cap E_2)$.

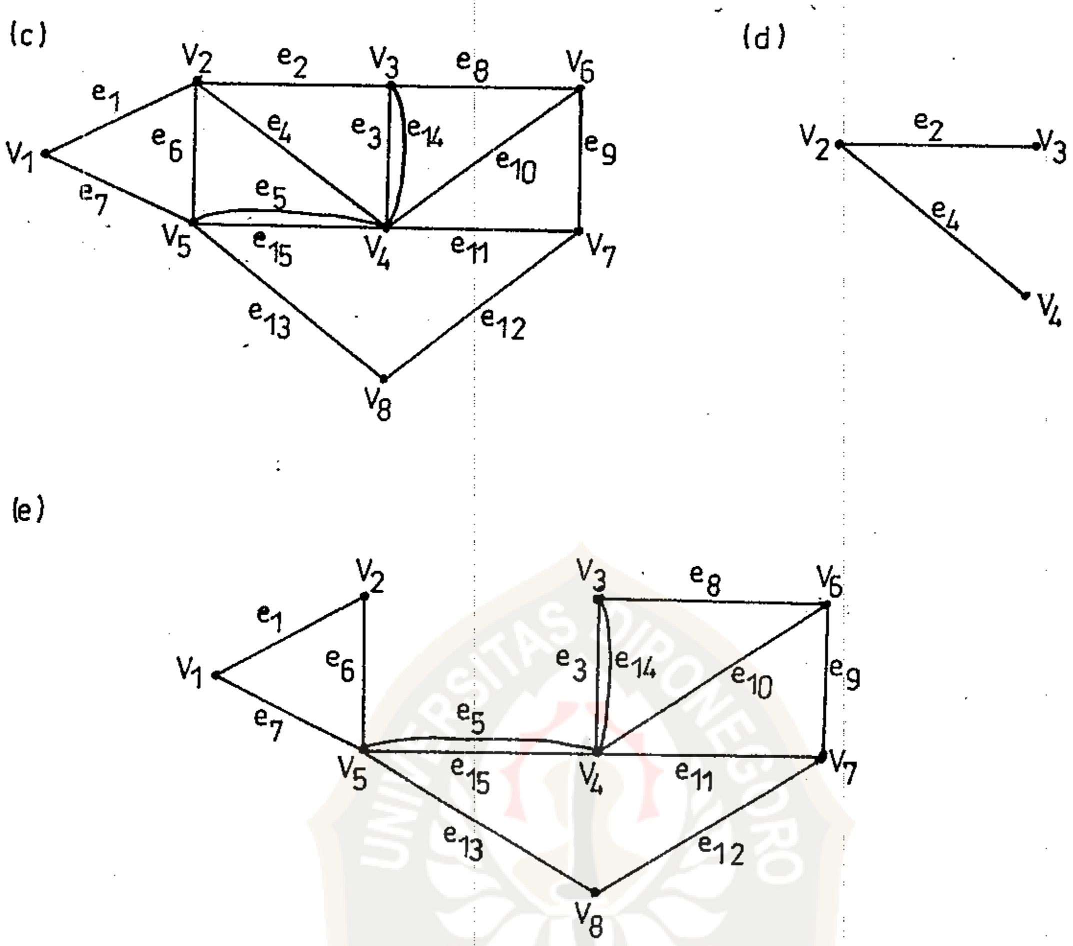
Contoh :

(a)



(b)





Gambar 2.1.9
(a) Graph G_1 ; (b) Graph G_2 ; (c) Graph G_3 ; (d) Graph G_4 ; (e) Graph G_5

2.1.3 Sifat - sifat Operasi dalam Graph

1. $G_1 \cup G_2 = G_2 \cup G_1$

2. $G_1 \cap G_2 = G_2 \cap G_1$

3. $G_1 \oplus G_2 = G_2 \oplus G_1$

4. $G \cup G = G \cap G$

5. $G_1 \oplus G_2 =$ Suatu null graph

2.2 Tree

2.2.1 Pengertian Tree

Definisi 2.2.1

Spanning subgraph adalah suatu subgraph dari graph G yang mengandung semua titik graph G.

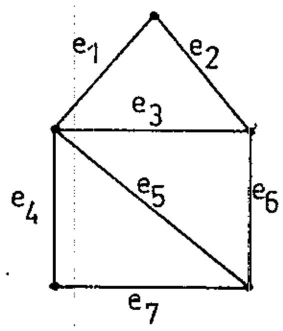
Definisi 2.2.2

Suatu spanning subgraph dari suatu graph dikatakan suatu **tree** bila dan hanya bila terhubung dan tidak mempunyai sirkuit.

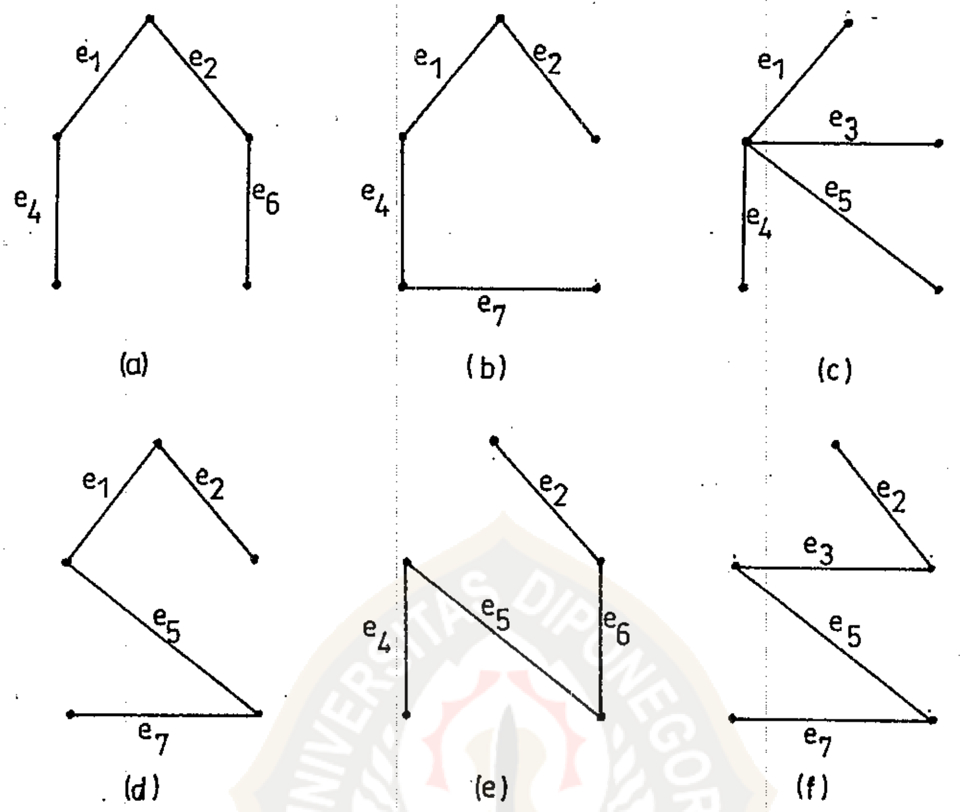
Suatu garis dari suatu tree disebut **cabang (branch)**.

Contoh :

Sebagian tree dari graph G pada gambar 2.2.1 ditunjukkan dalam gambar 2.2.2.



Gambar 2.2.1
Suatu graph yang digunakan untuk ilustrasi variasi subgraph



Gambar 2.2.2
Tree-tree dari graph pada gambar 2.2.1

Theorema 2.2.1

Misalkan G adalah suatu tree dengan sejumlah $n \geq 2$ titik, maka G mempunyai paling sedikit satu titik akhir.

Bukti :

Hal ini akan dibuktikan secara kontradiksi. Misalkan yang terjadi hal sebaliknya, jadi derajat tiap titik pada G paling sedikit dua. Mulai dari titik x_1, misalkan titik x_2 terhubung oleh x_1, karena derajat x_2 minimum dua maka x_2 juga terhubung pada titik lain yaitu x_3 yang tidak sama dengan x_1. Demikian juga x_3 terhubung pada x_4 yang tidak sama dengan x_2. Bila hal ini dilanjutkan maka pada akhirnya hal ini akan kembali pada salah satu titik yang telah disebutkan sebelumnya sehingga akan membentuk suatu sirkuit pada graph tersebut. Jadi graph tersebut bukan tree. Terjadi kontradiksi karena diketahui G adalah suatu tree. Pengandaian salah, yang benar G mempunyai paling sedikit satu titik akhir.

Theorema 2.2.2

Suatu tree dengan n titik akan mempunyai n-1 cabang.

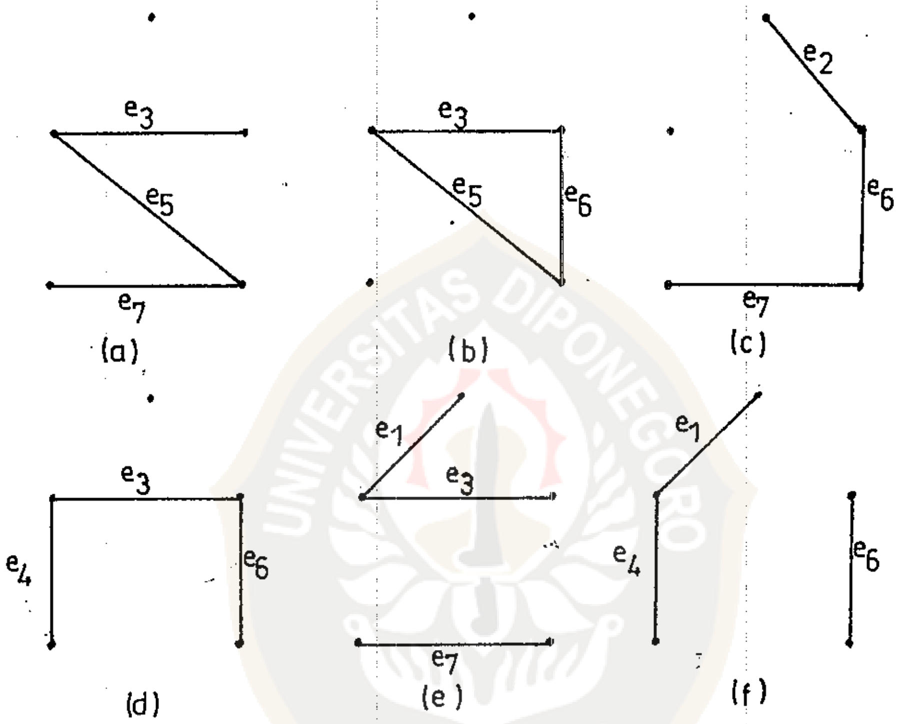
Bukti :

Hal ini akan dibuktikan secara induksi matematika. Jelas bahwa theorema ini berlaku untuk n=1. Andaikan benar untuk n=k, akan dibuktikan benar untuk n=k+1. Misalkan graph G adalah tree dengan k+1 buah titik. Menurut theorema 2.2.1, maka G pasti akan mempunyai satu titik akhir x dan e adalah cabang yang bertemu dengan x. Misalkan G' adalah graph baru yang diperoleh dari G dengan menghapus titik x dan cabang e, jelas bahwa graph G' ini adalah graph terhubung juga. Karena G' tidak mempunyai sirkuit yang terhubung maka dapat disimpulkan G' adalah suatu tree. Dan karena jumlah titik dari G' adalah (k+1)-1=k, maka menurut langkah induksi G' mempunyai k-1 cabang. Jadi G akan mempunyai (k-1)+1=k cabang. Bukti selesai.

Definisi 2.2.3

Komplemen dari suatu tree dalam suatu jaringan disebut suatu **cotree**.Suatu garis dalam cotree disebut **chord**.

Contoh :



Gambar 2.2.3
Cotree-cotree dari graph pada gambar 2.2.1

Theorema 2.2.3

Untuk suatu n titik dan e garis yang menghubungkan graph G, maka suatu tree berisi n-1 cabang dan suatu cotree berisi e-n+1 chord.

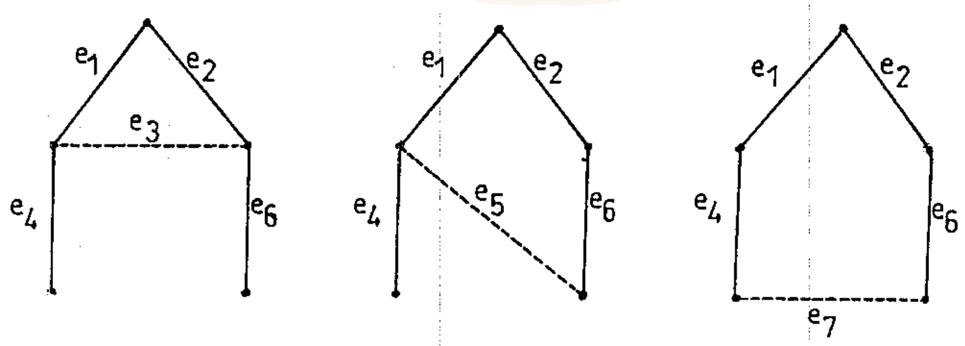
Bukti :

Dari theorema 2.2.2 telah dibuktikan bahwa untuk suatu tree dengan n titik akan berisi n-1 cabang, untuk bukti suatu cotree berisi e-n+1 chord adalah sebagai berikut. Karena chord adalah garis yang tidak termasuk dalam cabang G, maka jumlah chord adalah jumlah garis e dalam graph G dikurangi cabang tree dalam G sehingga didapat e-(n-1)=e-n+1. Jadi cotree dalam graph G berisi e-n+1 chord.

Dari sini ada suatu path tunggal (unique) terhubung antara suatu pasangan titik dalam suatu tree. Penambahan suatu chord untuk tree akan menghasilkan suatu sirkuit tunggal dalam graph hasil. Sirkuit ini disebut sirkuit fundamental, disingkat f-sirkuit.

Contoh :

Pada gambar 2.2.3a chord e_3, e_5 dan e_7 membentuk f-sirkuit untuk tree $t=e_4e_1e_2e_6$ pada gambar 2.2.2a. Pada gambar 2.2.4 ditunjukkan f-sirkuit yang diberikan oleh $e_1e_2e_3$ dan $e_1e_2e_6e_5$ dan $e_4e_1e_2e_6e_7$.



Gambar 2.2.4
Sirkuit fundamental dalam graph pada gambar 2.2.1 dari tree $t=e_1e_2e_4e_6$

Definisi 2.2.4

Suatu titik dikatakan terisolasi jika titik tersebut tidak terhubung dengan titik lainnya.

Definisi 2.2.5

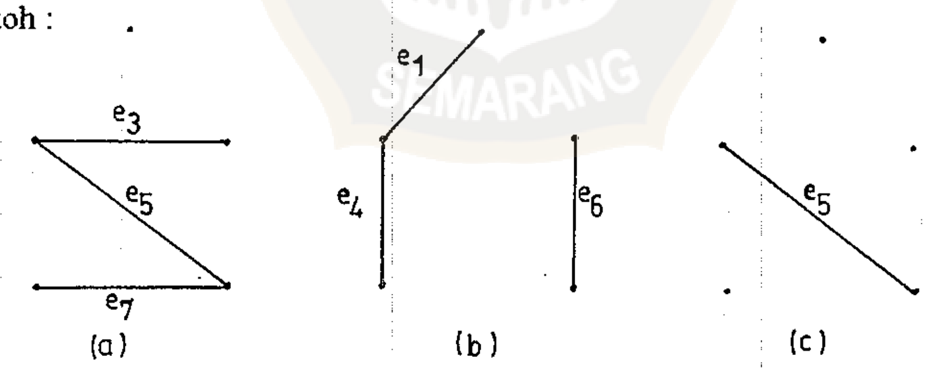
Suatu spanning subgraph dari suatu graph dikatakan suatu forest jika spanning subgraph tersebut tidak berisi sirkuit.

Satu atau beberapa komponennya mungkin masing-masing terdiri dari suatu titik terisolasi.

Definisi 2.2.6

Suatu komponen dari suatu forest dari suatu graph disebut suatu subtree.

Contoh :



Gambar 2.2.5
Forest dari graph pada gambar 2.2.1

Forest dalam gambar 2.2.5a dan gambar 2.2.5b masing-masing terdiri dari dua komponen sedangkan forest dalam gambar 2.2.5c terdiri dari empat komponen.

Tiap-tiap komponen dari forest dalam gambar 2.2.5 adalah suatu subtree. Maka suatu titik terisolasi dipandang sebagai suatu subtree.

Definisi 2.2.7

Jarak antara dua tree t_i dan t_j dari suatu graph adalah banyaknya garis-garis yang dimuat dalam tree t_i tetapi tidak dalam tree t_j.

Contoh :

Pandang tree $t_1=e_4e_1e_2e_6$, $t_2=e_2e_1e_5e_7$ dan $t_3=e_2e_3e_5e_7$ pada gambar 2.2.2. Tree t_1 dan t_2 berjarak 2 karena t_1 memuat dua garis e_4 dan e_6 yang tidak termuat dalam t_2 atau t_2 memuat dua garis e_5 dan e_7 yang tidak termuat dalam t_1. Tree t_2 dan t_3 berjarak 1 karena t_2 memuat satu garis e_2 yang tidak termuat dalam t_3 atau t_3 memuat satu garis e_3 yang tidak termuat dalam t_2. Sedangkan t_1 dan t_3 berjarak 3 karena t_1 memuat tiga garis e_1, e_4 dan e_6 yang tidak termuat dalam t_3 atau t_3 memuat tiga garis e_3, e_5 dan e_7 yang tidak termuat dalam t_1.

Definisi 2.2.8

Transformasi tree elementary adalah suatu operasi $e_1 \cup t\text{-}e_2 = t^*$ dengan t suatu tree dari suatu graph G dan e_1 suatu garis pada G tetapi tidak berada dalam t dan t* adalah suatu tree dari G dengan e_2 adalah suatu garis dari t.

Theorema 2.2.4

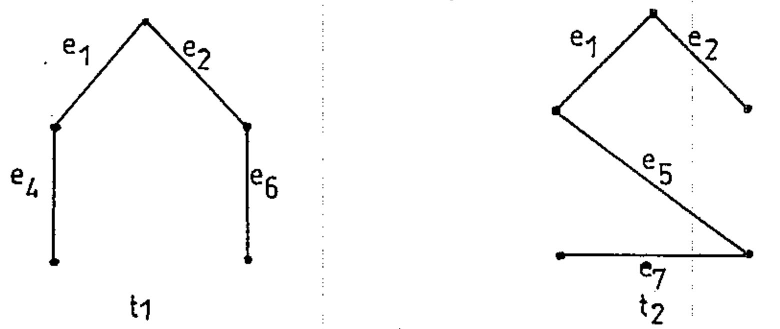
Setiap tree dari graph terhubung bisa diperoleh satu dari beberapa yang lain dengan urutan terbatas dari transformasi tree elementary.

Bukti :

Ambil t_1 dan t_2 adalah dua tree dari suatu graph terhubung G. Jika $t_1 \neq t_2$ maka terdapat cabang e_2 dalam t_2 tetapi tidak dalam t_1. Dari sini tidak semua garis dari f-sirkuit L dalam $t_1 \cup e_2$ didefinisikan dengan chord e_2 dari t_1 bisa dalam t_2, ada suatu cabang e_1 dari t_1 dalam L tetapi tidak dalam t_2. Pandang tree $t^* = e_2 \cup t_1 - e_1$, yang lebih tertutup ke t_2 daripada ke t_1. Sehingga jika $t^* \neq t_2$ proses diulang. Dari sini t_2 berisi suatu jumlah garis-garis terbatas, akhirnya akan dicapai t_2 dengan suatu urutan terbatas dari transformasi tree elementary.

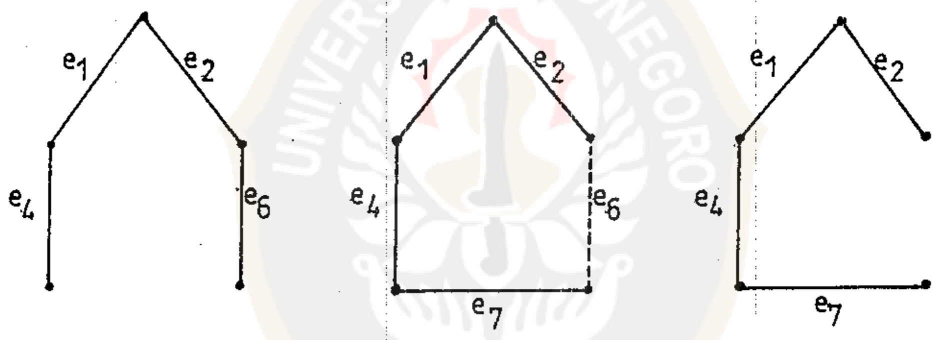
Contoh :

Diberikan dua tree t_1 dan t_2 dari gambar 2.2.1 seperti pada gambar 2.2.6. Akan ditunjukkan bahwa t_2 dapat diperoleh dengan dua transformasi tree elementary.

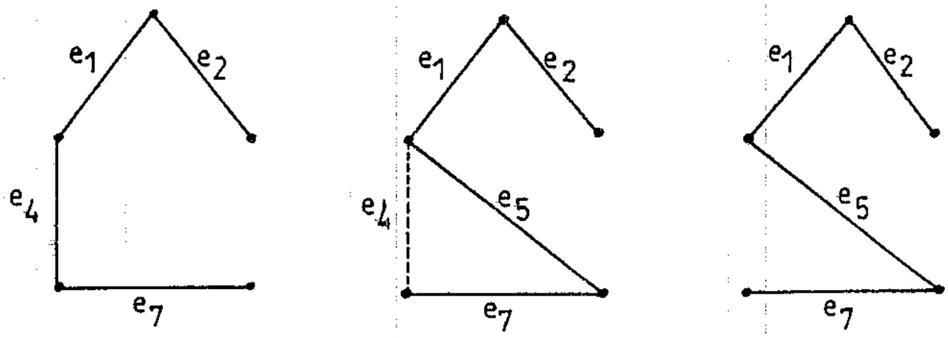


Gambar 2.2.6
Dua tree dari graph pada gambar 2.2.1a t_1 dan gambar 2.2.1d t_2

Garis e_7 dalam t_2 tetapi tidak dalam t_1. Transformasi tree elementary nya adalah $e_7 \cup t_1 - e_6 = e_2e_1e_4e_7$, hasil dalam suatu tree $t^*=e_2e_1e_4e_7$ dari G berjarak 1 ke t_2. Mengingat t_1 berjarak 2 ke t_2 maka t^* lebih tertutup ke t_2 daripada ke t_1. Dengan suatu transformasi tree elementary t_1 diubah ke t^* seperti ditunjukkan dalam gambar 2.2.7. Sekarang lihat bahwa e_5 dalam t_2 tetapi tidak dalam t^*, maka transformasi tree elementary kedua adalah $e_5 \cup t^* - e_4 = e_2e_1e_5e_7$ yang menghasilkan tree t_2 yang diinginkan. Operasi transformasi tree elementary kedua digambarkan pada gambar 2.2.8.



Gambar 2.2.7
Suatu transformasi tree elementary mengkonversi t_1 ke t^*



Gambar 2.2.8
Suatu transformasi tree elementary mengkonversi t^* ke t_2

2.2.2 Tree Maksimum

Suatu jaringan tak berarah adalah suatu graph bobot tak berarah dengan himpunan titik V dan himpunan garis E. Setiap garis (i,j) telah dikawankan dengan suatu bilangan real l(i,j) yang menyatakan panjang dari garis (i,j). Fungsi l dari himpunan garis E menyatakan fungsi panjang. Suatu jaringan tak berarah dapat ditulis sebagai G(V,E,l). Karena G(V,E,l) tak berarah, maka (i,j)=(j,i) untuk setiap $(i,j) \in E$.

Panjang dari suatu tree t dari G(V,E,l) adalah penjumlahan dari panjang garis cabang-cabang.

$$l(t) = \sum_{(i,j) \in t} l(i,j) \qquad (2.2.1)$$

Di antara semua tree t dari G(V,E,l) ada satu yang terpanjang yang disebut tree maksimum dari G.

$$l(t_{max}) = \max_{t} l(t) \qquad (2.2.2)$$

Demikian juga di antara semua tree t dari G(V,E,l) ada satu yang terpendek yang disebut tree minimum dari G.

$$l(t_{min}) = \min_{t} l(t) \qquad (2.2.3)$$

Theorema 2.2.6

Suatu tree dari jaringan tak berarah G(V,E,l) adalah maksimum jika dan hanya jika pertidaksamaan

$$l(x_1,x_k) \leq \min\{l(x_1,x_2), l(x_2,x_3), \ldots, l(x_{k-1},x_k)\} \qquad (2.2.4)$$

terpenuhi untuk setiap chord (x_1,x_k) yang sesuai dengan tree, dengan path tunggal antara titik x_1 dan x_k dalam tree adalah $(x_1,x_2)(x_2,x_3) \ldots (x_{k-1},x_k)$.

Bukti :

Syarat Perlu

Ambil t yang merupakan tree maksimum dari G. Jika pertidaksamaan (2.2.4) tidak terpenuhi maka ada satu cabang (x_i,x_{i-1}) , $1 \leq i \leq k-1$, dalam path tree tunggal terlihat bahwa

$$l(x_1,x_k) > l(x_i,x_{i-1}) \qquad (2.2.5)$$

Maka transformasi tree elementary

$$(x_1,x_k) \cup t - (x_i,x_{i-1}) = t^* \qquad (2.2.6)$$

akan menghasilkan suatu tree t* yang lebih panjang daripada t. Kontradiksi dengan asumsi bahwa t adalah suatu tree maksimum. Sehingga pertidaksamaan (2.2.4) terpenuhi untuk setiap chord (x_1,x_k) yang sesuai dengan tree t.

Syarat Cukup

Pertama ditunjukkan bahwa apabila t_1 dan t_2 adalah dua tree dari G yang masing-masing memenuhi pertidaksamaan (2.2.4), maka t_1 dan t_2 adalah tree dengan panjang yang sama. Kemudian t_1 dan t_2 dibagi dalam tiga kelas. Garis yang hanya dimiliki oleh t_1 disebut garis t_1, garis yang hanya dimiliki oleh t_2 disebut garis t_2 dan untuk garis-garis yang dimiliki oleh t_1 dan t_2 disebut garis bersama (common edge). Andaikan t_1 dan t_2 berbeda, ambil (x_1,x_k) suatu garis dalam t_2 dan pandang f-sirkuit

$$(x_1,x_k)(x_1,x_2)(x_2,x_3) \ldots (x_{k-1},x_k) \tag{2.2.7}$$

dibentuk oleh chord (x_1,x_k) dan path tree tunggal $P_{1k}(t_1) = (x_1,x_2)(x_2,x_3) \ldots\ldots (x_{k-1},x_k)$ antara titik x_1 dan x_k dalam t_1 yang tidak semua dapat menjadi garis bersama. Sebaliknya kondisi (2.2.7) akan menjadi sirkuit dari t_2 yang tidak mungkin untuk suatu tree. Sehingga ada garis-garis t_1 dalam $P_{1k}(t_1)$. Pertidaksamaan (2.2.4) menunjukkan bahwa setiap $(x_\alpha ,x_{\alpha+1})$ dari garis-garis t_1 ini mempunyai panjang terkecil $l(x_1,x_k)$. Kemudian ditunjukkan bahwa suatu panjang terkecil dari garis-garis t_1 itu adalah panjang $l(x_1,x_k)$. Andaikan bahwa setiap $(x_\alpha ,x_{\alpha+1})$ dari garis-garis t_1 ini mempunyai panjang yang lebih besar daripada $l(x_1,x_k)$. Kemudian f-sirkuit dibentuk oleh chord $(x_\alpha ,x_{\alpha+1})$ dan path tree tunggal $P_{\alpha\,(\alpha+1)}\,(t_2)$ antara titik x_α dan $x_{\alpha+1}$ dalam t_2 tidak berisi garis (x_1,x_k) pada saat t_2 memenuhi pertidaksamaan (2.2.4). Ini menunjukkan bahwa jika setiap $(x_\alpha ,x_{\alpha+1})$ garis t_1 ditempatkan dalam kondisi (2.2.7) dengan path tunggal $P_{\alpha\,(\alpha+1)}\,(t_2)$ dalam t_2 dihasilkan suatu subgraph t_2 yang berisi suatu sirkuit, terjadi suatu kontradiksi. Oleh karena itu paling sedikit ada satu garis dalam kondisi (2.2.7) yang mempunyai panjang yang sama dengan $l(x_1,x_k)$. Ambil $(x_\alpha ,x_{\alpha+1})$ suatu garis t_1 dengan $l(x_\alpha ,x_{\alpha+1}) = l(x_1,x_k)$. Maka transformasi tree elementary-nya adalah

$$(x_1,x_k) \cup t - (x_\alpha, x_{\alpha+1}) = t^* \tag{2.2.8}$$

akan menghasilkan suatu t* yang memiliki panjang yang sama dengan t_1. Lebih jauh lagi jarak antara t* dengan t_2 lebih pendek daripada jarak antara t_1 dengan t_2 dan hipotesa pertidaksamaan (2.2.4) sekali lagi terpenuhi untuk t*. Dari sini argumen tersebut dapat diulang yang menunjukkan bahwa t_1 dapat ditransformasi ke t_2 oleh suatu barisan terbatas dari transformasi tree elementary yang memiliki suatu rangkaian panjang tree-tree yang sama, dengan hasil akhir adalah t_2. Hal ini menunjukkan bahwa t_1 dan t_2 memenuhi pertidaksamaan (2.2.4), maka t_1 dan t_2 memiliki panjang yang sama. t_2 merupakan suatu tree yang maksimum sebab memenuhi pertidaksamaan (2.2.4), secara tidak langsung setiap tree t_1 yang memenuhi pertidaksamaan (2.2.4) merupakan tree maksimum.

2.2.3 Algoritma Tree Maksimum

Di sini akan digunakan suatu teknik penambahan sederhana yang dikenal dengan metode Greedy, yang membentuk suatu tree maksimum dengan memasukkan garis-garis yang panjang dan mengeluarkan garis-garis yang pendek sampai suatu tree maksimum terbentuk. Metode Greedy ini cukup umum digunakan untuk masalah tree maksimum. Di bawah ini akan dijelaskan tentang *Algoritma Kruskal* untuk menentukan tree maksimum yang merupakan variasi dari metode Greedy.

Diberikan suatu jaringan tak berarah G(V,E,l), suatu proses pewarnaan garis dijelaskan sebagai berikut. Pada awalnya semua garis dari G tidak diwarnai, kemudian suatu garis diwarnai dengan warna biru atau merah sesuai dengan aturan-aturan berikut:

- *Aturan Biru*

Pilih suatu potongan tanpa garis-garis biru. Di antara garis-garis yang tidak diwarnai dalam potongan itu, pilih satu yang mempunyai panjang maksimum dan warnai garis tersebut dengan warna biru.

- *Aturan Merah*

Pilih suatu sirkuit tanpa garis-garis merah. Di antara garis-garis yang tidak diwarnai pada sirkuit itu, pilih satu yang mempunyai panjang minimum dan warnai garis tersebut dengan warna merah.

Metode Greedy akan mewarnai semua garis dari G dan ketika semua garis diwarnai, yang biru membentuk suatu tree maksimum. Sehingga pewarnaan biru menandakan penerimaan dari suatu garis dan pewarnaan merah berarti penolakan.

Algoritma Kruskal

Algoritma ini pertama diusulkan oleh Kruskal (1956) yang merupakan variasi dari metode Greedy. Untuk melaksanakan algoritma ini pertama harus mengurutkan garis-garis dari G(V,E,l) ke dalam urutan tanpa pengurangan dari panjang dan menggambarkan subtree biru dari forest, maka dapat diuji apakah dua titik dalam subtree biru yang sama.

Langkah-langkah Algoritma Kruskal :

Step 1 : Pilih satu garis terpanjang dari G(V,E,l) dan warnai dengan biru. Tentukan i=0.

Step 2: Di antara garis-garis yang belum dipilih, pilih satu garis (x,y) terpanjang. Jika titik x dan y kepunyaan subtree biru yang sama, warnai (x,y) dengan merah. Jika sebaliknya warnai (x,y) dengan biru.

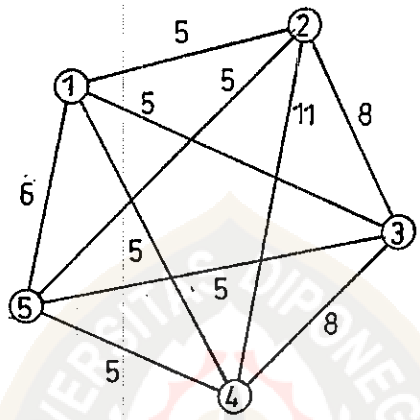
Step 3 : Tentukan i=i+1.

Step 4 : Jika i=m-1, hentikan. Dengan m adalah jumlah garis dari G. Jika belum, kembali ke step 2.

Dengan kata lain, algoritma ini dimulai dengan memilih satu garis terpanjang dari G. Pada tiap step berturut-turut dipilih suatu garis terpanjang di antara garis-garis yang belum dipilih dan tidak terbentuk suatu sirkuit dengan garis-garis yang sudah dipilih. Sebagai alternatif lain, dapat dimulai dengan suatu graph lengkap yang menghilangkan suatu garis pada suatu saat. Di antara garis yang beluim dihilangkan, hilangkan satu garis terpendek yang tidak akan memutuskan graph tersebut.

Contoh :

Diberikan suatu jaringan G(V,E,l) seperti gambar 2.2.11. Aplikasikan algoritma Kruskal untuk menentukan suatu tree maksimum dari jaringan tersebut.



Gambar 2.2.11
Graph lengkap berbobot lima titik

Pertama disusun garis-garis dari G dalam urutan tanpa pengurangan dari panjang garis. Garis-garis diurutkan dari yang terpanjang sampai yang terpendek sebagai berikut : (2,4),(3,4),(2,3),(1,5),(1,2),(1,3),(1,4),(2,5),(3,5),(4,5).

Langkah-langkahnya adalah :

Step 1 : Pilih garis (2,4) dan warnai biru. Tentukan i=0.

Step 2 : Pilih garis (3,4) dan warnai biru.

Step 3 : Tentukan i=1.

Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (2,3) dan warnai merah.

Step 3 : Tentukan i=2.

Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (1,5) dan warnai biru.

Step 3 : Tentukan i=3.

Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (1,2) dan warnai merah.

Step 3 : Tentukan i=4.

Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (1,3) dan warnai merah.

Step 3 : Tentukan i=5.

Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (1,4) dan warnai merah.

Step 3 : Tentukan i=6.

Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (2,5) dan warnai biru.

Step 3 : Tentukan i=7.

Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (3,5) dan warnai merah.

Step 3 : Tentukan i=8.

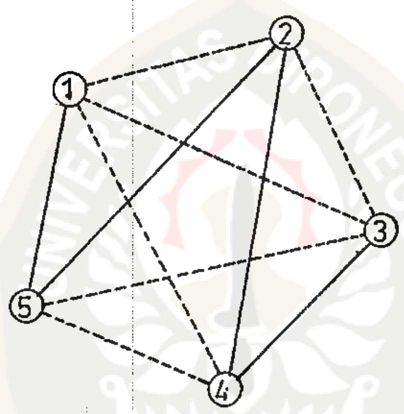
Step 4 : Kembali ke step 2.

Step 2 : Pilih garis (4,5) dan warnai merah.

Step 3 : Tentukan i=9.

Step 4 : Stop.

Setelah batas dari proses pewarnaan, jaringan warna akhir digambarkan dalam gambar 2.2.12 dengan garis utuh menunjukkan garis biru dan garis putus-putus menunjukkan garis merah. Dan akan dihasilkan suatu tree maksimum yang ditunjukkan dengan garis utuh.



Gambar 2.2.12
Suatu tree maksimum

2.3 Potongan i-j (i-j cut)

Definisi 2.3.1

Simbol (X_p,X_q) menunjukkan himpunan semua garis yang dihubungkan dari titik anggota X_p ke titik anggota X_q.

Contoh :

Suatu graph terhubung dengan $V = \{v_1,v_2,v_3,v_4,v_5,v_6\}$, seperti pada gambar 2.3.1. Jika dipilih $X = \{v_1,v_2,v_6\}$ dan $\overline{X} = \{v_3,v_4,v_5\}$ maka akan didapat :

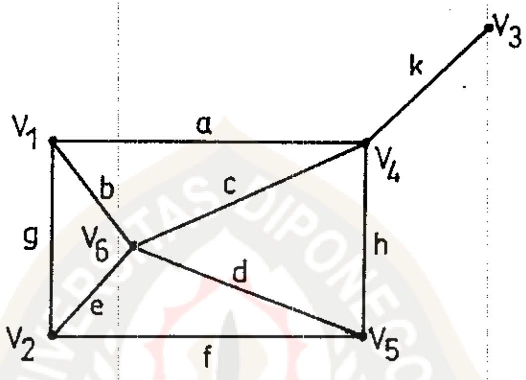
$(\overline{X}, X) = \{b,e,g\}$

$(\overline{X}, \overline{X}) = \{h,k\}$

$(X, \overline{X}) = \{a,c,d,f\}$

Sehingga :

$(V,V) = \{a,b,c,d,e,f,g,h,k\}$.



Gambar 2.3.1
Graph terhubung G dengan 6 titik

Pandang tiga subgraph (X,X), $(X,\overline{X})$, $(\overline{X},\overline{X})$ dengan $X=V-\overline{X}$ dari suatu graph lengkap G_c. Ambil e suatu garis yang berbeda dengan G_c dengan titik akhir p dan q, maka :

1. Andaikan kedua titik p dan q dalam X maka e adalah dalam subgraph (X,X) tetapi tidak dalam $(X,\overline{X})$ dan $(\overline{X},\overline{X})$.

2. Andaikan kedua titik p dan q dalam $\overline{X}$ maka e adalah dalam $(\overline{X},\overline{X})$ tetapi tidak dalam dua subgraph yang lain.

3. Andaikan titik p dalam X dan titik q dalam $\overline{X}$ maka e adalah dalam $(X,\overline{X})$ tetapi tidak dalam dua subgraph yang lain.

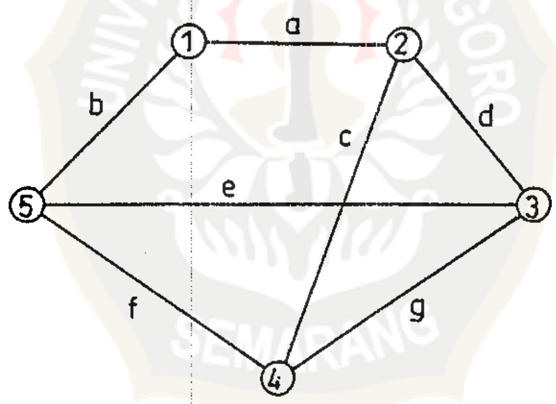
Sehingga dapat ditunjukkan bahwa setiap garis dalam G_c adalah tepat berada dalam satu subgraph, yaitu (X,X) atau $(X,\overline{X})$ atau $(\overline{X},\overline{X})$.

Definisi 2.3.2

Untuk dua titik i dan j yang berbeda dari suatu graph G, suatu potongan i-j adalah himpunan garis $(X,\overline{X})$ dari G dengan $i \in X$ dan $j \in \overline{X}$, dimana X adalah subset dari V dan $\overline{X} = V - X$.

Contoh :

Diketahui suatu graph terhubung G seperti pada gambar 2.3.2.



Gambar 2.3.2
Suatu graph terhubung G

Dari gambar 2.3.2, potongan $v_1 - v_3$ adalah suatu himpunan garis $(X,\overline{X}) = \{a,e,f\}$ dengan $v_1 \in X$ dan $v_3 \in \overline{X}$.

Himpunan garis $\{a,b\},\{b,c,d\},\{d,e,g\},\{a,c,e,g\}$ juga merupakan potongan v_1-v_3.

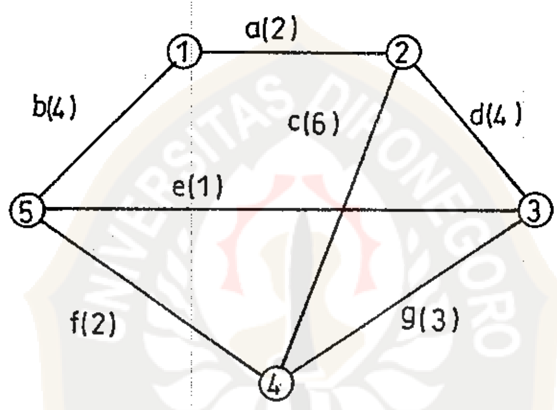
Definisi 2.3.3

Kapasitas $(X, \overline{X})$ yaitu kapasitas dari suatu potongan i-j dalam suatu jaringan komunikasi G(V,E,c,f) yang didefinisikan dengan :

$$c(X, \overline{X}) = \sum_{(x,y)\in(X,\overline{X})} c(x, y)$$

dengan c(x,y) adalah kapasitas garis (x,y).

Contoh 2.3.3 :



Gambar 2.3.3
Graph terhubung dengan kapasitas garis

Dari gambar 2.3.2, misalkan setiap garis diberi suatu kapasitas seperti pada gambar 2.3.3 di atas, maka himpunan garis - himpunan garis berikut merupakan potongan 1-3 dengan kapasitas :

- {a,b} dengan kapasitas c(1,2)+c(1,5) = 2+4 = 6
- {b,c,d} dengan kapasitas c(1,5)+c(2,4)+c(2,3) = 4+6+4 = 14
- {d,e,g} dengan kapasitas c(2,3)+c(5,3)+c(3,4) = 4+1+3 = 8
- {a,e,f} dengan kapasitas c(1,2)+c(5,3)+c(5,4) = 2+1+2 = 5
- {a,c,e,g} dengan kapasitas c(1,2)+c(2,4)+c(5,3)+c(3,4) = 2+6+1+3 = 12

Definisi 2.3.4

Suatu potongan minimum i-j (C_{min}) adalah suatu potongan i-j dengan kapasitas minimum di antara semua potongan i-j, didefinisikan dengan :

$$c(C_{min}) = \min_i \{c(X_i, \overline{X_i})\}$$

dengan $(X_i, \overline{X}_i)$ adalah potongan i-j dalam G.

Contoh 2.3.4 :

Dari contoh 2.3.3 di atas :

potongan minimum 1-3 = min [c{a,b},c{b,c,d},c{d,e,g},c{a,e,f},c{a,c,e,g}]

= min [6,14,8,5,12]

= 5

2.4 Matriks Kapasitas Terminal

2.4.1 Pengertian Matriks Kapasitas Terminal

Definisi 2.4.1

Diberikan pasangan titik i dan j dari suatu jaringan komunikasi G(V,E,c,f). Harga minimum potongan i-j adalah suatu potongan i-j dengan kapasitas minimum di antara semua potongan i-j, yang tidak lain adalah kapasitas terminal $\tau_{ij} = \min [c(X_p, \overline{X}_p)]$ dengan $(X_p, \overline{X}_p)$ adalah potongan i-j dari G.

Untuk jaringan komunikasi tidak berarah G(V,E,c,f) pada gambar 2.3.3, dari contoh 2.3.4, harga dari potongan $(X, \overline{X})$ yang merupakan potongan minimum 1-3 adalah 5, harga tersebut merupakan elemen τ_{13}. Demikian juga untuk elemen τ_{23}, potongan minimum 2-3 adalah :

$$\tau_{23} = c(Y, \overline{Y}) = c(\{1,2,4,5\},3)$$

$$= c(2,3)+c(3,5)+c(3,4) = 4+1+3 = 8$$

Dengan cara yang sama akan didapatkan kapasitas terminal dari setiap pasangan titik-titik sebagai berikut :

$$\tau_{12} = \tau_{21} = 5 \qquad \tau_{24} = \tau_{42} = 11$$

$$\tau_{13} = \tau_{31} = 5 \qquad \tau_{25} = \tau_{52} = 5$$

$$\tau_{14} = \tau_{41} = 5 \qquad \tau_{34} = \tau_{43} = 8$$

$$\tau_{15} = \tau_{51} = 6 \qquad \tau_{35} = \tau_{53} = 5$$

$$\tau_{23} = \tau_{32} = 8 \qquad \tau_{45} = \tau_{54} = 5$$

Dalam suatu sistem komunikasi, kapasitas terminal τ_{ij} mewakili kapasitas yang tersedia untuk komunikasi dari i ke j. Kemampuan komunikasi antara semua pasangan titik dalam G dapat disajikan sebagai suatu matriks yang disebut *matriks kapasitas terminal.*

Definisi 2.4.2

Matriks kapasitas terminal $T=[\tau_{ij}]$ dari suatu jaringan komunikasi G(V,E,c,f) merupakan suatu matriks bujursangkar berderajat n, dengan :

- Elemen τ_{ij} (elemen baris ke i kolom ke j, $i \neq j$) adalah kapasitas terminal dari i ke j.
- Elemen-elemen diagonal τ_{ii} didefinisikan infinite.

Bentuk dari matriks kapasitas terminal adalah sebagai berikut :

$$T = \begin{array}{cc} & \begin{array}{cccc} 1 & 2 & \dots & n \end{array} \\ \begin{array}{c} 1 \\ 2 \\ . \\ n \end{array} & \begin{bmatrix} \infty & \tau_{ij} & \dots & \tau_{1n} \\ \tau_{21} & \infty & \dots & \tau_{2n} \\ . & . & \dots & . \\ \tau_{n1} & \tau_{n2} & \dots & \infty \end{bmatrix} \end{array}$$

Langkah-langkah membentuk matriks kapasitas terminal adalah sebagai berikut :

1. Menentukan kapasitas terminal τ_{ij} dari semua pasangan titik-titiknya.
2. Membentuk suatu matriks bujursangkar berderajat n dan memasukkan kapasitas terminal τ_{ij} sebagai elemen-elemen matriks tersebut sesuai dengan baris dan kolomnya dan diagonal utamanya ∞ .

Sehingga matriks kapasitas terminal dari jaringan pada gambar 2.3.3 adalah sebagai berikut :

$$T = \begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} \infty & 5 & 5 & 5 & 6 \\ 5 & \infty & 8 & 11 & 5 \\ 5 & 8 & \infty & 8 & 5 \\ 5 & 11 & 8 & \infty & 5 \\ 6 & 5 & 5 & 5 & \infty \end{bmatrix} \end{array}$$

2.4.2 Jaringan Ekuivalen Tree

Theorema 2.4.1

Suatu matriks real simetri $T=[\tau_{ij}]$ dengan elemen-elemen non negatif dan infinite sepanjang diagonal utamanya adalah matriks kapasitas terminal dari jaringan komunikasi tak berarah bila dan hanya bila untuk setiap i,j dan k

$$\tau_{ij} \geq \min [\tau_{ik}, \tau_{kj}] \qquad (2.4.1)$$

Bukti :

Syarat Perlu

Ambil G(V,E,c,f) jaringan komunikasi tak berarah. $(X, \overline{X})$ adalah potongan minimum i-j dari G, maka $\tau_{ij} = c(X, \overline{X})$.

Jika titik $k \in X$ maka $(X, \overline{X})$ juga suatu potongan k-j dan

$$\tau_{ij} = c(X, \overline{X}) \geq \tau_{kj} \qquad (2.4.2)$$

dan pertidaksamaan (2.4.1) terpenuhi.

Jika titik $k \in \overline{X}$ maka $(X, \overline{X})$ juga suatu potongan i-k dan

$$\tau_{ij} = c(X, \overline{X}) \geq \tau_{ik} \qquad (2.4.3)$$

dan pertidaksamaan (2.4.1) terpenuhi.

Syarat Cukup

Susun graph lengkap tak berarah G(V,E) yang mempunyai n titik dan pada setiap garis i-j diberikan harga τ_{ij} yang dipandang sebagai panjang dari garis tersebut. Ambil t_{max} suatu tree maksimum dalam jaringan hasil G(V,E,l) dengan $l(i,j) = \tau_{ij}$.

Dari theorema 2.2.6 bahwa apabila $(i,x_1)(x_1,x_2)$ ... $(x_{u-1},x_u)(x_u,j)$ adalah path tunggal antara titik i dan j dalam t_{max} maka :

$$l(i,j) \leq \min\,[l(i,x_1), l(x_1,x_2), \ldots, l(x_{u-1},x_u), l(x_u,j)] \qquad (2.4.4)$$

atau

$$\tau_{ij} \leq \min\,[\tau_{ix1}, \tau_{x1x2}, \tau_{x2\,x3}, \ldots, \tau_{xu\text{-}1\,xu}, \tau_{xu\,j}] \qquad (2.4.5)$$

Dengan asumsi pertidaksamaan (2.4.1) terpenuhi untuk semua i,j dan $k \in V$ maka

$$\tau_{ij} \geq \min\,[\tau_{ix1}, \tau_{x1j}]$$

$$\tau_{ij} \geq \min\,[\tau_{ix1}, \tau_{x1x2}, \tau_{x2\,j}]$$

$$\tau_{ij} \geq \min\,[\tau_{ix1}, \tau_{x1x2}, \tau_{x2\,x3}, \tau_{x3\,j}]$$

.

.

.

$$\tau_{ij} \geq \min \left[\tau_{ix1}, \tau_{x1x2}, \tau_{x2\,x3}, \ldots, \tau_{xu-1\,xu}, \tau_{xu\,j} \right] \qquad (2.4.6)$$

Kombinasi dari pertidaksamaan (2.4.5) dan (2.4.6) menunjukkan bahwa :

$$\tau_{ij} = \min \left[\tau_{ix1}, \tau_{x1x2}, \tau_{x2\,x3}, \ldots, \tau_{xu-1\,xu}, \tau_{xu\,j} \right] \qquad (2.4.7)$$

Oleh karena itu jika setiap cabang (x,y) dari t_{max} sekarang ditetapkan sebagai kapasitas c(x,y) = l(x,y), dan jika setiap chord dari cotree t_{max} dihilangkan dari G, maka diperoleh suatu jaringan komunikasi tak berarah t_{max} (V,E_t,c) dengan panjang setiap cabangnya merupakan kapasitas terminal dari T.

Suatu konsekuensi penting dari pembuktian ini adalah jika T dapat dicapai ini dinyatakan dengan suatu tree n titik berisi n-1 cabang, matriks T dapat mempunyai paling banyak n-1 elemen bilangan yang berbeda.

Definisi 2.4.3

Dua jaringan komunikasi n titik merupakan aliran ekuivalen, atau tepatnya ekuivalen jika mempunyai matriks kapaistas terminal yang sama.

Akibat 2.4.1

Setiap jaringan komunikasi tak berarah adalah ekuivalen terhadap suatu tree dan terdapat paling banyak n-1 bilangan kapasitas terminal yang berbeda.