BAB III
ALGORITMA TRIANG DAN ALGORITMA COLOR

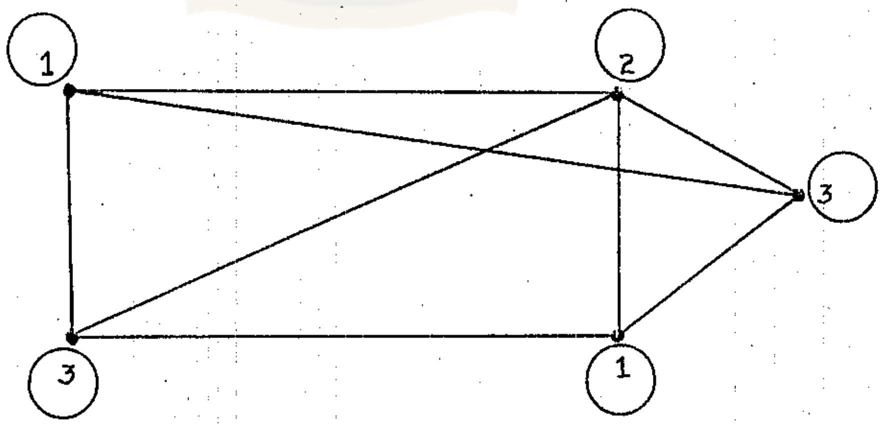
3.1 BILANGAN KROMATIK

Pewarnaan simpul (vertek) pada suatu Graph adalah pemberian warna pada semua simpul dari graph tersebut, sedemikian sehingga tidak terdapat dua simpul saling berdampingan yang mempunyai warna yang sama.

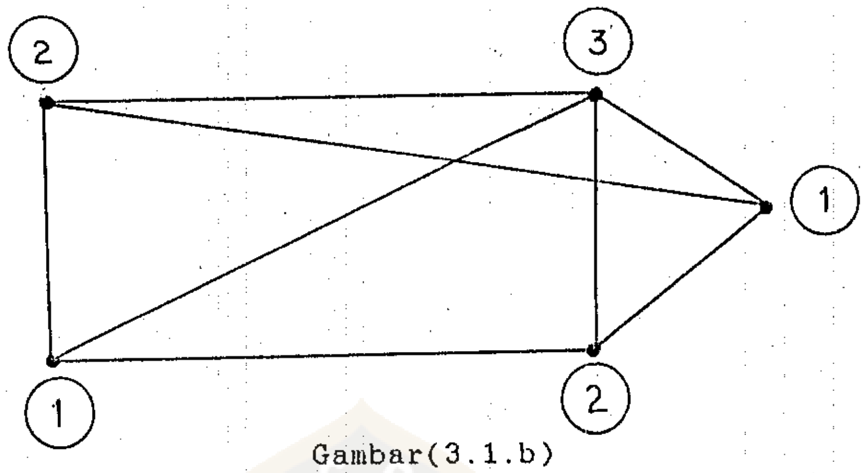
Pewarnaan k dari graph G adalah pemberian k warna yang berbeda pada simpul-simpul dari G. Sebuah Graph dikatakan berwarna k jika graph tersebut mempunyai pewarnaan k.

Bilangan kromatik x(G) dari graph G adalah jumlah minimum k sehingga G berwarna k.

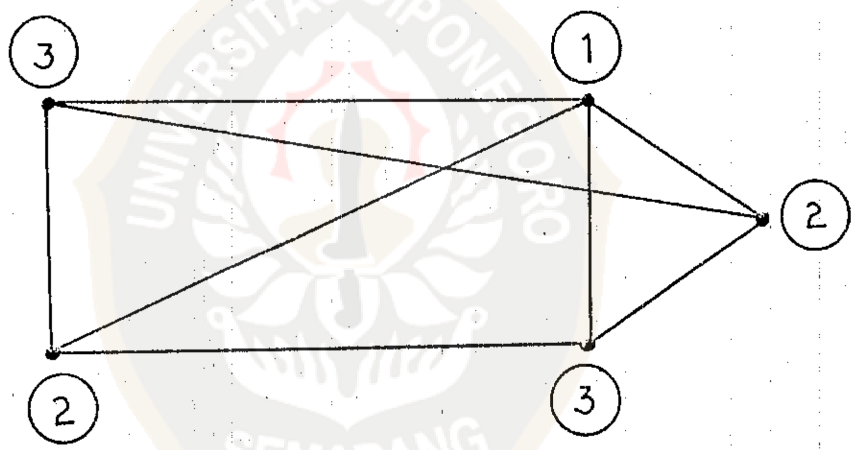
Graph G dikatakan k-kromatik jika x(G)=k. Sebagai contohnya bilangan kromatik dari Graph pada gambar (3.1) di bawah ini :



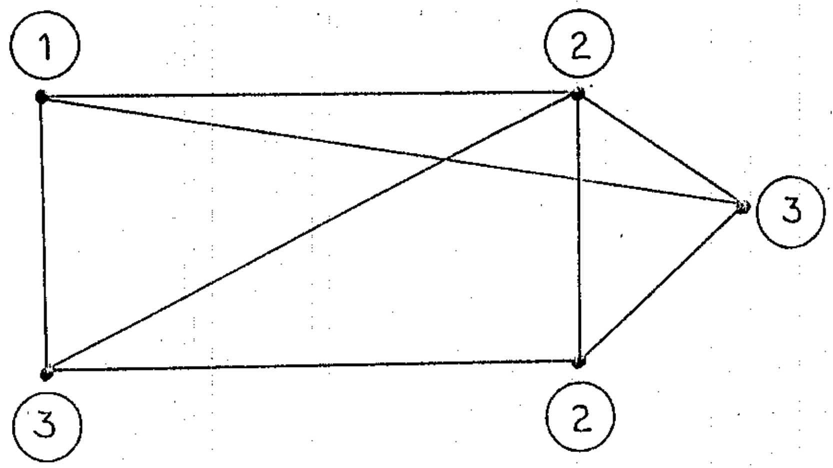
Gambar(3.1.a)



Gambar(3.1.b)



Gambar(3.1.c)



Gambar(3.1.d)

*) KETERANGAN :

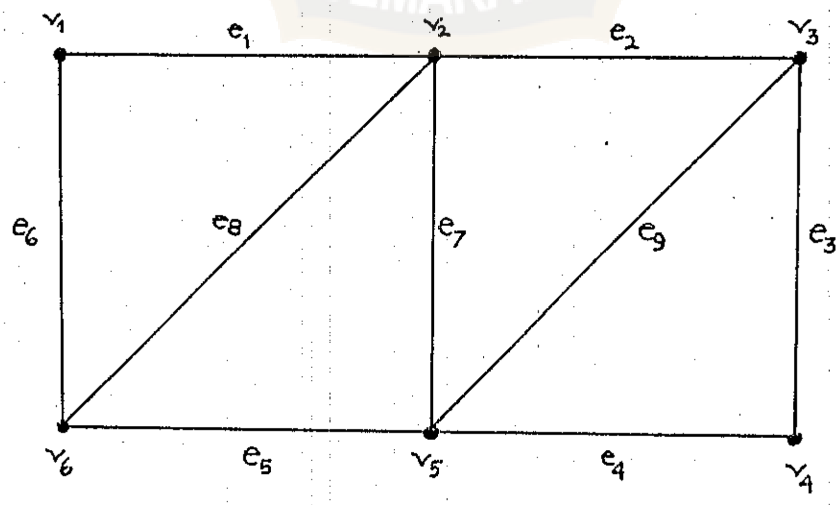
Pada gambar (3.1.a),(3.1.b),(3.1.c) merupakan pewarnaan graph dengan bilangan kromatik adalah 3. Sedangkan untuk gambar (3.1.d) *bukan* pewarnaan graph, karena simpul 2 dengan simpul 2 saling berdampingan.

3.2 TRIANGULATED GRAPH

DEFINISI 3.1 :

Suatu graph G dikatakan *triangulated* jika setiap *sirkuit* dari G yang panjangnya ≥ 4 mempunyai paling sedikit sebuah *chord*.

Sebagai contoh dari *triangulated graph* pada gambar (3.2) yaitu :



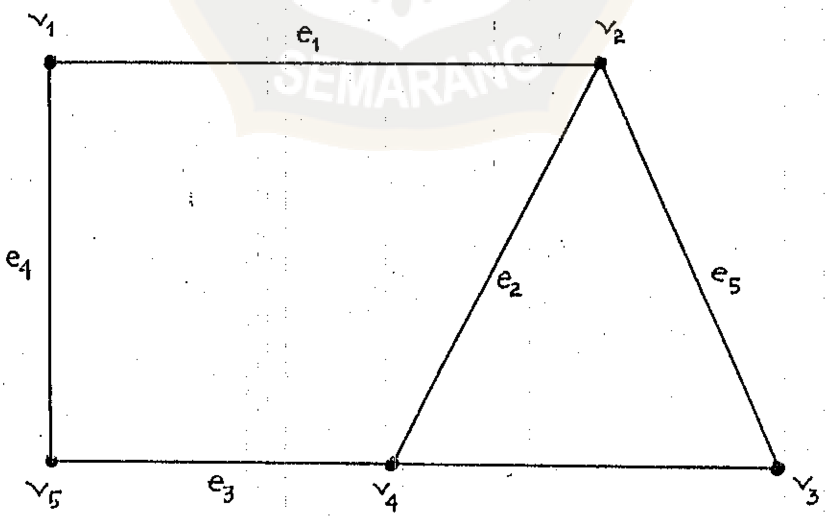
Gambar(3.2)

*) KETERANGAN GAMBAR :

Sirkuit-sirkuit dari graph G pada gambar (3.2) adalah $C_1=\{e_1,e_7,e_5,e_6\}$; $C_2=\{e_2,e_3,e_4,e_7\}$; $C_3=\{e_1,e_2,e_9,e_5,e_6\}$ dan $C_4=\{e_1,e_2,e_3,e_4,e_5,e_6\}$.

Panjang sirkuit C_1 adalah 4 dan banyaknya chord pada C_1 adalah 1 yaitu e_8. Pada C_2, panjang sirkuitnya adalah 4 dan banyaknya chord pada C_2 adalah 1 yaitu e_9. Dan pada C_3, panjang sirkuitnya adalah 5, banyaknya chord pada C_3 adalah 2 yaitu e_8 dan e_7. Sedangkan panjang sirkuit pada C_4 adalah 6 dengan jumlah chordnya sama dengan 3 yaitu e_7, e_8, e_9.

Contoh Graph yang tidak *Triangulated,* yaitu :



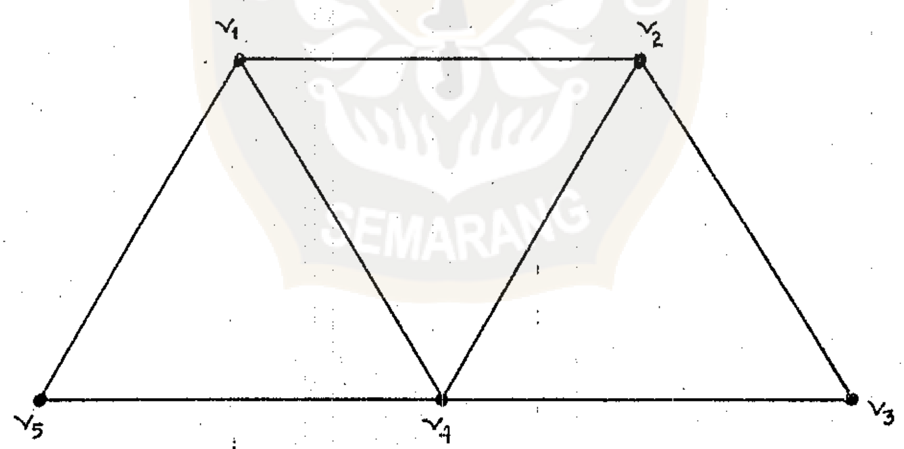
Gambar(3.3)

Pada gambar diatas, tidak *triangulated* karena kita dapat menemukan sirkuit dengan panjang 4(empat) yaitu $C=\{e_1, e_2, e_3, e_4\}$, tetapi tidak mempunyai *chord* satupun.

DEFINISI 3.2 :

Simpul v dari graph G di katakan *simplicial* jika semua simpul yang adjacent dengan v saling adjacent satu dengan yang lainnya. Hal ini berarti bahwa v termasuk dalam satu klik.

Contoh :



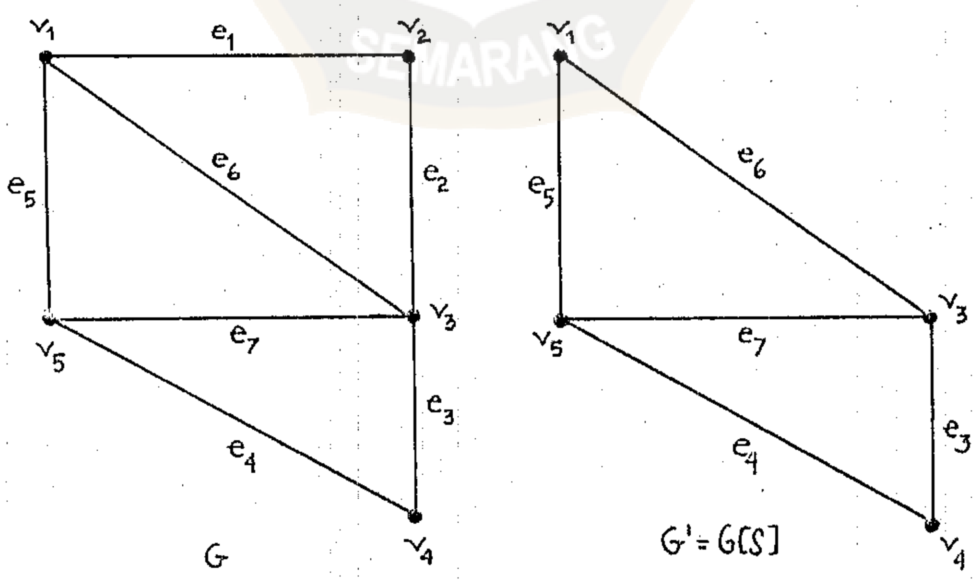
Gambar(3.4)

Pada gambar (3.4) diatas, simpul v_3 dan v_5 adalah *simplicial*, karena simpul-simpul yang adjacent dengan v_3 yaitu v_2 dan v_4 ; serta yang adjacent dengan v_5 yaitu v_1 dan v_4 , saling adjacent dengan yang lain.

DEFINISI 3.3 :

Barisan simpul $v_1,\ldots,v_n$ dari graph G dikatakan *sempurna* jika untuk $k = 1,\ldots,n$, v_k simplicial dalam G.

Kalau kita perhatikan bahwa setiap *triangulated graph* mempunyai sedikitnya 2 simpul simplicial. Lebih lanjut suatu graph G(V,E) adalah *triangulated* jika dan hanya jika V mempunyai barisan yang sempurna. Jika graph G *triangulated* maka setiap subgraph G terinduksi juga *triangulated* dan G mempunyai sebanyak-banyaknya |v| klik serta *kardinalitas dari klik maksimum sama dengan pewarnaan minimum.*



Gambar(3.5)

Pada gambar (3.5) diatas, graph G dengan 5 simpul dan 7 ruas adalah *triangulated* dan mempunyai 2 simpul simplicial yaitu v_2 dan v_4. Barisan sempurna dalam G yaitu { v_2,v_4 }. Graph G′ juga *triangulated.*

Pewarnaan minimum dari graph G adalah 3 dan hal ini sama dengan klik maksimum dari graph G tersebut.

Kemudian gunakan aturan diatas untuk membentuk *Triangulated Subgraph Terinduksi Maksimal* G(T) dari graph G, dan mencari klik maksimum dari G(T).

Algoritma ini disebut *Algoritma TRIANG*, yang merupakan salah satu dari dua hal yang penting dari prosedur yang akan di gunakan dalam mencari klik maksimum. Kemudian yang kedua *Algoritma COLOR*, yaitu untuk mencari pewarnaan minimum dari G(T), disini kardinality *k* dari pewarnaan minimum sama dengan ukuran dari klik maksimum, lalu G(T) di perluas dengan menambahkaan simpul padanya sambil mempertahankan bilangan kromatiknya, sehingga menjadi *k-kromatik* subgraph terinduksi maksimal dari G.

Andaikan subgraph ini adalah G(W). Jika G(W) = G, maka klik maksimum yang di dapat dalam G(T) adalah klik maksimum dari G.

3.3 ALGORITMA TRIANG

Untuk menentukan apakah graph G=(V,E) *triangulated* dapat dikerjakan secara lebih efisien dengan memeriksa apakah G mengandung barisan yang sempurna.

Hal ini dapat di lakukan dengan mengerjakan sebanyak n-kali untuk $k=1,\ldots,n$ dengan langkah sebagai berikut :

Cari simpul simplicial dalam $G(V - \{w_1,\ldots,w_{k-1}\})$.

Jika tidak ada, berhenti. G tidak *triangulated.*

Jika ada, ambil simpul lain w_k dan jadikan $k \longrightarrow k+1$.

Jika $k+1 = n$, berhenti. $\{w_1,\ldots,w_n\}$ adalah barisan yang sempurna.

Ulangi langkah di atas.

DEFINISI 3.4 :

Diberikan barisan $\sigma = (v_1,..v_n)$ dari V, sebut elemen dari himpunan :

$N^+(v_i) := \{v_j \in V - \{v_i\} \mid (v_i,v_j) \in E \text{ dan } j > i\}$

Sebagai suksesor dari v_i , dan sebut suksesor dari v_i dengan indeks terkecil sebagai suksesor pertama dari v_i.

DEFINISI 3.5 :

v_i dikatakan *quasi-simplicial* dalam $\sigma = (v_1,\ldots,v_n)$. Jika *suksesor pertama* dari v_i adjacent dengan setiap suksesor lainnya .

TEOREMA 3.1 :

Diberikan graph $G = (V,E)$ dan barisan $\sigma = (v_1,\ldots,v_n)$.
pernyataan berikut adalah ekivalen ;
(i) untuk $i = 1,\ldots,n$, v_i simplicial dalam $G(\{v_1,\ldots,v_n\})$
(ii) untuk $i = 1,\ldots,n$, v_i quasi-simplicial dalam σ

BUKTI :

(i)$\longrightarrow$(ii) jelas . Jika v_i simplicial dalam G maka simpul yang adjacent dengan v_i saling adjacent satu dengan yang lainnya . Ini berarti v_i quasi-simplicial dalam σ .
Sekarang misalkan (ii) terpenuhi, maka v_n simplicial dalam $G(\{v_n\})$. Anggap v_i simplicial dalam

$G(\{v_1,\ldots,v_n\})$ untuk i = $n, n-1, \ldots, k+1$, dan misalkan $i = k$.

Jika v_k mempunyai kurang dari dua suksesor , maka (ii)——>(i) terpenuhi ; Jika tidak , misalkan $w_1,\ldots w_p$ adalah suksesor dari v_k adjacent dengan $w_1 = v_1$ sebagai suksesor pertama.

Karena v_k quasi-simplicial , $w_2,\ldots w_p$ adjacent dengan $w_1 = v_1$. Karena v_1 diasumsikan simplicial $w_2,\ldots w_p$ adjacent secara berpasangan .

Maka v_k simplicial , karena $i=k$ maka berarti pula v_i simplicial dalam $G(\{v_1,\ldots,v_n\})$.

AKIBAT :

Barisan $\sigma = (v_1,\ldots,v_n)$ adalah sempurna jika dan hanya jika untuk i = $1,\ldots,n$, v_i adalah quasi-simplicial dalam σ.

ALGORITMA TRIANG

Andaikan i menyatakan nomor urutan dalam σ yang akan ditandai , T himpunan simpul yang termasuk dalam triangulated sub-graph , dan $\sigma(T)$ adalah barisan yang didefinisikan dalam T :

0. Inisialisasi

Tandai label ϕ pada setiap simpul .

Jadikan $i \longrightarrow n$, $T \longrightarrow \phi$, $\sigma(T) \longrightarrow \phi$

1. Pemilihan v .

Jika $i = 0$, berhenti : G(T) adalah triangulated subgraph terinduksi maksimal . Jika tidak , pilih sebuah simpul tak bernomor v dengan label terbesar , tandai bilangan i pada v , jadikan $v_i \longrightarrow v$, dan lanjutkan ke langkah 2.

2. Memeriksa apakah v dapat ditambahkan pada T .

Misalkan σ^o adalah barisan yang diperoleh dari $\sigma(T)$ dengan menambahkan v sebagai elemen pertama. Jika v adalah quasi-simplicial dalam σ^o,jadikan $T \longrightarrow T \cup \{v\}$, $\sigma(T) \longrightarrow \sigma^o$ dan lanjutkan ke langkah 3 .Jika tidak , jadikan $i \longrightarrow i - 1$ dan ulangi ke langkah 1 .

3. Pelabelan .

Tambahkan i pada label dari simpul tak bernomor w yang adjacent dengan v , jadikan $i \longrightarrow i-1$ dan pergi ke langkah 1.

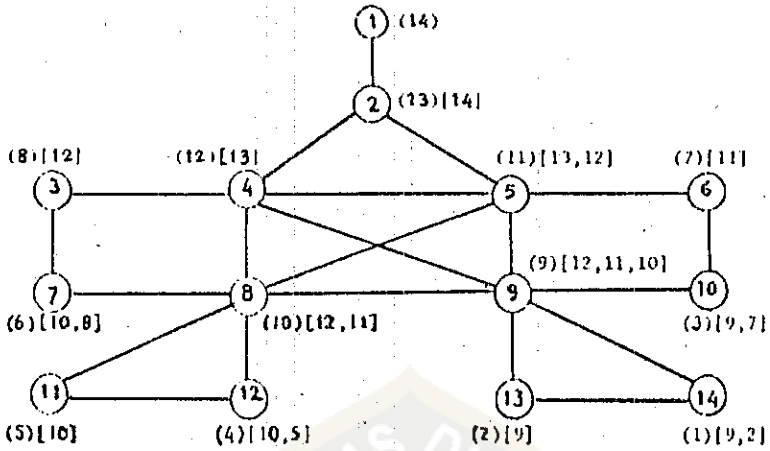
Sebagai contoh ilustrasi algoritma TRIANG , yaitu :

Misalkan G adalah graph seperti gambar (3.6.a) , dimana nomer urut dari setiap simpul ditunjukkan dalam tanda kurung, sedangkan label-labelnya ditunjukan dalam kurung siku.

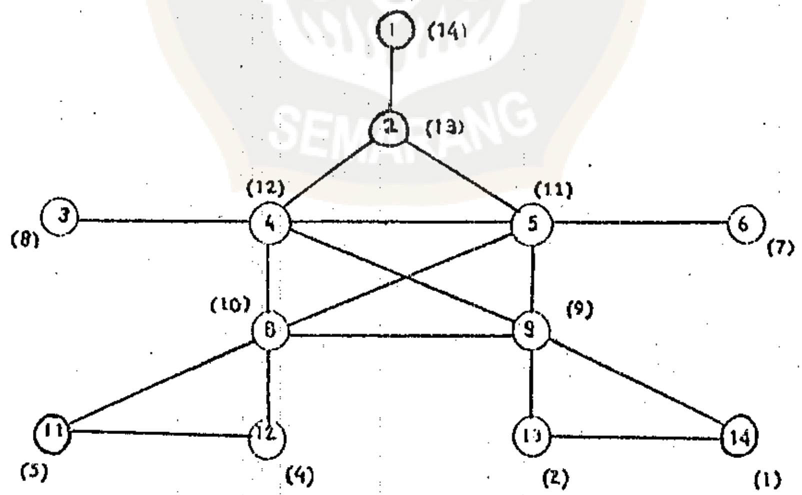
Setelah menandai label dari setiap simpul dengan ϕ (tidak di tunjukkan dalam gambar), dipilih simpul 1 (pemilihan simpul secara sembarang) dan tandai dengan nomor 14. Jelas, simpul 1 adalah quasi-simplicial di G({1}), dan kemudian jadikan T={1}, σ(T)=(1). Tambahkan 14 pada label dari simpul satu-satunya yang adjacent dengan simpul 1, namakan simpul 2. Pada iterasi berikutnya dipilih simpul 2, tandai simpul itu dengan nomor 13, lalu jadikan T={1,2}, σ(T)=(2,1), dan tambahkan 13 pada label dari simpul 4 dan simpul 5. Kemudian dipilih simpul 4 (pemilihan dilakukan berdasarkan urutan dari bilangan), lalu nomori simpul itu 12, jadikan T={1,2,4}, σ(T)=(4,2,1), dan tambahkan 12 pada label dari simpul 3,5,8,9. Pada iterasi berikutnya, simpul 5 menjadi nomor 11, dan 11 ditambahkan pada label dari simpul 6,8,9. Kemudian simpul 8 menjadi nomor 10, dan 10 di tambahkan pada label dari simpul 7,9,11,12. Simpul 9 menjadi bernomor 9 dan 9 ditambahkan pada label-label dari simpul 10,13,dan 14. Berikutnya simpul 3 menjadi bernomor 8, dan 8 di tambahkan pada label dari simpul 7. Simpul 6 menjadi benomor 7, kemudian 7 di tambahkan pada label dari simpul 10.

Disini simpul berikutnya dengan label terbesar secara leksikography adalah 7 : simpul 7 menjadi bernomor 6, tetapi dengan menetapkan σ^{o}= (7,6,3,9,8,5,4,2,1) pada langkah 2, terlihat bahwa simpul 7 tidak quasi-simplicial dalam σ^{o}; suksesor pertama dari simpul 7 yaitu simpul 3 tidak adjacent dengan simpul 8 yang juga merupakan suksesor dari simpul 7. Maka simpul 7 di tolak (tidak termasuk dalam T).

Kemudian iterasi dilanjutkan. Simpul 10 di tolak (setelah menjadi nomor 3) dan semua simpul yang tersisa di tambahakan pada T. Barisan akhir dari σ(T) adalah (14,13,12,11,6,3,9,8,5,4,2,1) dan TSTM dari G(T) di perlihatkan pada gambar (3.6.a). Simpul dengan jumlah suksesor terbesar dalam σ(T) adalah simpul 9, dan klik maksimum yang di dapat adalah himpunan simpul {9,8,5,4} dan ini dapat di lihat pada gambar (3.6.b).



Gambar (3.6.a)



Gambar (3.6.b)

TSTM yang di cari dengan menggunakan algoritma TRIANG adalah maksimum jika himpunan simpulnya hingga, tetapi belum tentu merupakan maksimal kardinal.

Dapat di catat bahwa langkah 1 dari algoritma TRIANG, kita bebas memilih simpul yang tak bernomor dengan label terbesar secara *leksikography*, dengan pemilihan yang berbeda akan menghasilkan TSTM yang berbeda pula.

3.4 ALGORITMA COLOR

Setelah TSTM dari G(T) terbentuk kemudian G(T) diperluas menjadi subgraph terinduksi maksimal G(W) dari G dengan ukuran klik maksimum yang sama dengan ukuran klik maksimum dari G(T). Hal itu dapat di lakukan dengan cara sebagai berikut : Karena G(T) triangulated, ukuran klik maksimum dari G(T) sama dengan bilangan kromatiknya, yaitu kardinality dari pewarnaan minimum (dari simpul) dari G(T).

Selanjutnya, di berikan barisan yang sempurna σ = $(v_1,\dots,v_{|T|})$ dari simpul-simpul G(T). Mencari pewarnaan minimum dalam G(T) secara berurutan :

Bentuk *k* kelompok warna, dimana *k* adalah ukuran dari klik maksimum dalam G(T), dan untuk memeriksa semua simpul dari G(T) dimulai dari urutan yang paling belakang, lalu tandai semua simpul dengan kelompok warna yang sesuai.

Karena setiap v_i mempunyai sebanyak-banyaknya (k-1) suksesor dalam $G(v_1,\ldots,v_T)$, setiap simpul akan sesuai dalam beberapa kelompok warna. Memperluas G(T) sambil mempertahankan k kelompok warna dapat di kerjakan secara sederhana dengan memeriksa setiap simpul $v \in V - T$ secara berurutan, dan menempatkan simpul-simpul tersebut dalam kelompok warna yang sesuai.

Berikut ini akan di berikan prosedur yang jelas dari hal di atas.

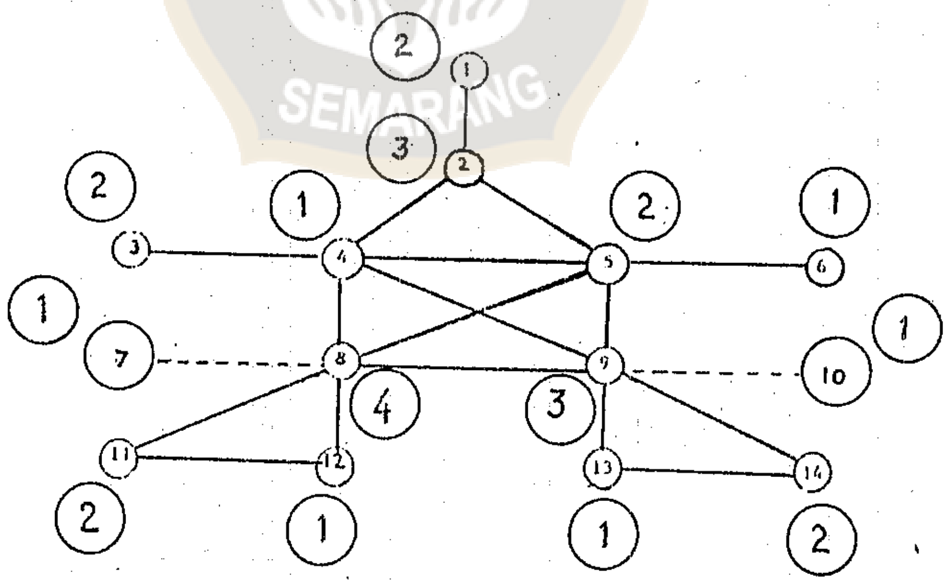
3.4.1 ALGORITMA COLOR 1.

0. Inisialisasi k kelompok warna $C_1,\ldots,C_k$. $C_j = \phi$, $j=1,\ldots,k$, jadikan $V \longrightarrow V - T$, dan pergi kelangkah 1.
1. Jika $T = \phi$, lanjutkan ke langkah 2. Jika tidak pilih simpul $v \in T$ dengan nomor urut terbesar dalam σ , dan pilih sebuah kelompok warna C_j yang tidak mengandung simpul yang adjacent dengan v, jadikan :
   $C_j \longrightarrow C_j \cup \{v\}$, $T \longrightarrow T - \{v\}$, lalu ulangi kelangkah 1.
2. Pilih simpul $v \in V$. Jika ada kelompok warna yang tidak mengandung simpul yang adjacent dengan v, sebut kelompok warna tersebut C_j dan jadikan :
   $C_j \longrightarrow C_j \cup \{v\}$.

3 Masukkan v kedalam kelompok warna C_j yang mengandung jumlah terkecil simpul-simpul yang adjacent dengan v dan dicoba untuk memindahkan setiap $w \in C_j$ yang adjacent dengan v kedalam kelompok warna lain yang tidak mengandung simpul yang adjacent dengan w.

Algoritma di atas memperluas TSTM dari G dengan klik maksimum berukuraan *k* ke *k*-kromatik subgraph terinduksi maksimal dari G.

Contoh, diambil graph TSTM dari gambar(3.6.b) diatas yaitu :



*) KETERANGAN :

Dari graph TSTM diatas, didapat klik dengan ukuran 4 (empat), berarti ada 4 (empat) kelompok warna yang akan diberikan pada graph TSTM tersebut. Ternyata semua simpul dapat diwarnai (ditandai) dengan 4 (empat) kelompok warna tersebut, sehingga proses selesai.

3.4.2 ALGORITMA COLOR 2.

Dalam prosedur yang akan di jelaskan berikut ini, kadang-kadang perlu untuk mencari *k*-kromatik triangulated subgraph terinduksi maksimal dari suatu graph yang belum mempunyai kardinal klik yang maksimal.

Dalam kasus ini kita memakai versi yang telah di modifikasi dari algoritma di atas, yang di sebut dengan Algoritma COLOR 2, yaitu :

0. Inisialisasi *k* kelompok warna $C_1, \ldots, C_k$. $C_j = \phi$, $j=1,\ldots,k$, jadikan $V \longrightarrow V - T$, dan lanjutkan ke langkah 1.
1. Pilih simpul $v \in V$. Jika ada kelompok warna yang tidak mengandung simpul yang adjacent dengan v, sebut kelompok warna tersebut C_j dan jadikan :

   $C_j \longrightarrow C_j \cup \{v\}$.

2. Masukkan v kedalam kelompok warna C_j yang mengandung jumlah terkecil simpul-simpul yang adjacent dengan v dan dicoba untuk memindahkan setiap $w \in C_j$ yang adjacent dengan v kedalam kelompok warna lain yang tidak mengandung simpul yang adjacent dengan w.

Jika langkah 3 (pada Algoritma COLOR 1) atau langkah 2 (pada Algoritma COLOR 2) berhasil untuk α simpul $v \in V-W$, kardinalitas dari w bertambah sebesar α.