MEMBANDINGKAN ALGORITMA *D'SATUR* DENGAN ALGORITMA *VERTEX MERGE* DALAM PEWARNAAN GRAF TAK BERARAH

Daratun Nasihin[1*] Endang Lily[2], M. D. H. Gamal[2]

[1] Mahasiswa Program Studi S1 Matematika
Jurusan Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Riau
Kampus Binawidya Pekanbaru (28293), Indonesia

*daratun.nasihin@yahoo.com

ABSTRACT

This article provides an undirected graph vertex coloring. Graph vertex coloring can be done by using the coloring algorithm, including *d'satur* algorithm and *vertex merge* algorithm. Of the two algorithms, a more efficient algorithm is determined that generates chromatic number $\chi(G)$ which is the minimum or produces fewer colors. This article is a review of the work of Handrizal et. al [Journal of Computer Science 7(5):664-670].

Keywords: *graph, graph vertex coloring, d'satur algorithm, vertex merge algorithm.*

ABSTRAK

Artikel ini membahas pewarnaan simpul graf tak berarah. Pewarnaan simpul garf dapat dilakukan dengan menggunakan algoritma pewarnaan, diantaranya algoritma *d'satur* dan algoritma *vertex merge*. Dari kedua algoritma, ditentukan algoritma yang lebih efisien yang menghasilkan bilangan kromatik $\chi(G)$ terkecil atau menghasilkan warna yang lebih sedikit. Kajian ini merupakan *review* sebagian dari karangan Handrizal et. al [Journal of Computer Science 7(5):664-670].

Kata kunci: *graf, pewarnaan simpul graf, algoritma d'satur, algoritma vertex merge.*

1. PENDAHULUAN

Teori graf merupakan salah satu bidang matematika yang diperkenalkan pertama kali oleh seorang ahli matematika asal Swiss, Leonhard Euler 1736 [3, h. 3]. Ide besarnya muncul sebagai upaya penyelesaian masalah jembatan Konigsberg. Dari permasalahan itu, akhirnya Euler mengembangkan beberapa konsep mengenai teori graf. Keunikan teori graf adalah kesederhanaan pokok bahasan yang dipelajari, karena dapat disajikan dengan simpul dan sisi. Teori graf telah banyak memberi masukan kepada ilmu baru salah satunya adalah pewarnaan graf.

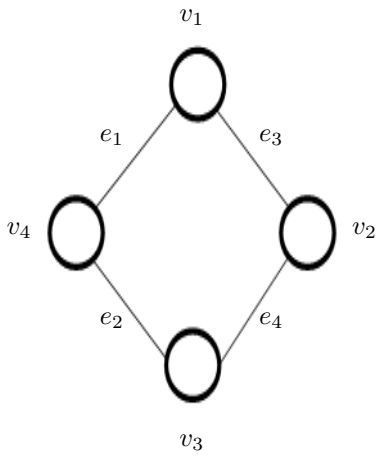
Ada tiga macam pewarnaan graf, yaitu pewarnaan sisi (*edge coloring*), pewarnaan simpul (*vertex coloring*) dan pewarnaan wilayah (*region coloring*). Pewarnaan sisi merupakan pemberian warna pada sisi graf sampai sisi-sisi yang saling berhubungan tidak memiliki warna yang sama. Pewarnaan simpul merupakan pemberian warna pada setiap simpul dimana warna yang sama akan diberikan pada simpul yang tidak saling bertetangga, sedangkan yang bertetangga akan diberi warna lain. Pewarnaan wilayah merupakan pemberian warna pada wilayah pada graf sehingga tidak ada wilayah yang bersebelahan memiliki warna yang sama.

Pewarnaan simpul graf merupakan salah satu subjek yang menarik dan terkenal dalam bidang graf. Teori-teori yang berhubungan dengan hal tersebut telah banyak dikembangkan, berbagai algoritma telah diperkenalkan diantaranya algoritma lebah koloni, Welch Powel, *d'satur*, *vertex merge* dan lain-lain. Artikel ini membahas tentang pewarnaan simpul graf dengan algoritma *d'satur* dan algoritma *vertex merge* kemudian membandingkan algoritma manakah yang lebih efisien dalam pewarnaan simpul graf.

2. GRAF

Pada bagian ini diberikan beberapa pengertian dasar dan beberapa definisi sebagai teori pendukung dalam pewarnaan simpul graf.

Teori graf merupakan salah satu bahasan dalam matematika diskrit yang menarik untuk dibahas karena berkaitan dengan permasalahan yang banyak ditemui dalam kehidupan sehari-hari. Graf G didefinisikan sebagai pasangan himpunan (V, E), ditulis dengan notasi $G = (V, E)$, dengan V adalah himpunan tidak kosong dari simpul-simpul dan E adalah himpunan sisi yang menghubungkan sepasang simpul. Selanjutnya, simpul-simpul dinotasikan dengan $\{v_1, v_2, v_3, ..., v_n\}$ dan $\{e_1, e_2, e_3, ..., e_m\}$ adalah sisi-sisi graf. Untuk lebih jelasnya, diberikan contoh graf $G = (V, E)$ dengan 4 simpul dan 4 sisi seperti tampak pada Gambar 1.



Gambar 1: Graf G

Pada Gambar 1, graf $G(V, E)$ memiliki 4 simpul dan 4 sisi yaitu $V = \{v_1, v_2, v_3, v_4\}$ dan $E = \{e_1, e_2, e_3, e_4\}$.

Sisi $e = (v_i, v_j)$ adalah sisi yang menghubungkan simpul v_i dan v_j. Jika $e = (v_i, v_j)$ adalah sisi pada graf G, maka v_i dan v_j disebut terhubung (*adjacent*), sedangkan v_i dengan e dan v_j dengan e disebut terkait (*incident*) [2, h. 1]

Definisi 1 [4, h. 372] Bilangan kromatik dari suatu Graf dilambangkan dengan $\chi(G)$, adalah jumlah warna minimum yang diperlukan untuk pewarnaan simpul dari suatu graf. Suatu graf G dengan k-kromatik jika $\chi(G) = k$.

Definisi 2 [4, h. 372] Suatu graf dikatakan *k-colorable* jika memiliki simpul tepat *k-coloring.*

3. MATRIKS KETETANGGAN

Misalkan G adalah graf dengan n simpul dan misalkan simpul telah diurutkan, katakanlah $\{v_i, v_j, ..., v_n\}$. Kemudian matrik ketetanggaan $A(G) = [a_{i,j}]$ dari graf G adalah matrik $m \times m$ yang didefinisikan dengan,

$$a_{i,j} = \begin{cases} 1 & \text{untuk } v_{i,j} \in E_G \\ 0 & \text{untuk } v_{i,j} \notin E_G \end{cases}$$

Berikut adalah matriks ketetanggaan A dari graf G pada Gambar 1 seperti terlihat pada Tabel 1.

Tabel 1: Matrik ketetanggaan

	v_1	v_2	v_3	v_4
v_1	0	1	0	1
v_2	1	0	1	0
v_3	0	1	0	1
v_4	1	0	1	0

Pada Tabel 1, matrik ketetanggaan beranggotakan 4 simpul. Apabila simpul v_i dan v_j terhubung maka diberi nilai 1, dan apabila simpul v_i dan v_j tidak terhubung maka diberi nilai 0.

4. ALGORITMA *D'SATUR*

Pada tahun 1979 Brelaz menemukan algoritma *d'satur* (*degree of saturation*). Algoritma *d'satur* [1] merupakan algoritma pewarnaan terurut dari derajat simpul tertinggi hingga derajat terendah. Derajat saturasi suatu simpul v_i adalah derajat simpul v_i dikurangi banyaknya warna berbeda yang sudah muncul pada tetangga v_i. Langkah-langkah konstruksi pewarnaan simpul-simpul dari graf G dengan menggunakan algoritma *d'satur* adalah sebagai berikut.

1. Susun simpul berdasarkan urutan jumlah derajat dari derajat simpul tertinggi hingga derajat simpul terendah.

2. Warnai simpul yang berderajat tertinggi dengan warna yang diinginkan.

3. Pilih simpul dengan derajat terbesar, jika ada yang sama, pilih simpul dengan derajat tertinggi dari subgraf yang belum diwarnai.

4. Warnai simpul yang dipilih dengan warna berikutnya.

5. Jika seluruh simpul sudah diwarnai, berhenti. Jika tidak, kembali ke langkah 3.

5. ALGORITMA *VERTEX MERGE*

Masalah perkiraan pewarnaan simpul adalah masalah yang banyak diteliti dan sejumlah algoritma juga telah tersedia. Salah satu algoritma tersebut adalah algoritma *vertex merge* [5]. Algoritma ini didasarkan pada deterministik, karena bertentangan dengan algoritma acak yang banyak diusulkan. Algoritma *vertex merge* menggunakan beberapa metode heuristik untuk memilih simpul yang akan diwarnai, heuristik yang digunakan oleh algoritma ini adalah untuk menemukan sub himpunan dari simpul dengan derajat tertinggi, yaitu simpul dengan jumlah perbedaan sisi terbesar dengan tetangga. Jika sub himpunan ini hanya berisi satu simpul, ia dipilih menjadi warna. Jika lebih dari satu simpul tetap di himpunan, pemilihan ini kemudian dibuat dalam urutan penurunan jumlah tetangga yang belum diwarnai. Berikut ini adalah langkah - langkah algoritma *vertex merge*.

1. Susun simpul berdasarkan pengurutan jumlah derajat.

2. Pilih simpul pertama yang belum diwarnai didalam himpunan.

3. Warnai simpul yang sudah dipilih dengan warna yang diinginkan.

4. Gabungkan simpul dengan simpul pertama yang tidak bertetangga.

5. Warnai simpul yang sudah dipilih dengan warna yang sama. Jika tidak ada lagi simpul yang tidak bertetangga, kembali ke langkah 2.

6. Jika seluruh simpul sudah diwarnai. Berhenti. Jika tidak, kembali ke langkah 2.

6. APLIKASI PEWARNAAN SIMPUL GRAF

Pada bagian ini diberikan sebuah aplikasi masalah pewarnaan simpul graf yang selanjutnya diselesaikan dengan masing-masing algoritma dan kemudian membandingkan hasil pewarnaan berdasarkan masing-masing algoritma.

Misalkan ada 12 jenis zat kimia yang perlu disimpan digudang. Beberapa pasang dari zat tersebut tidak dapat disimpan diruangan yang sama, karena campuran gasnya bersifat eksplosif (mudah meledak). Untuk zat-zat yang semacam itu perlu dibangun ruangan-ruangan terpisah yang dilengkapi ventilasi dan penyedot udara keluar yang berlainan. Jika lebih banyak ruangan yang digunakan maka akan membutuhkan biaya yang lebih besar untuk penyimpanan. Untuk itu perlu diketahui jumlah minimum ruangan yang diperlukan untuk menyimpan zat-zat kimia tersebut.

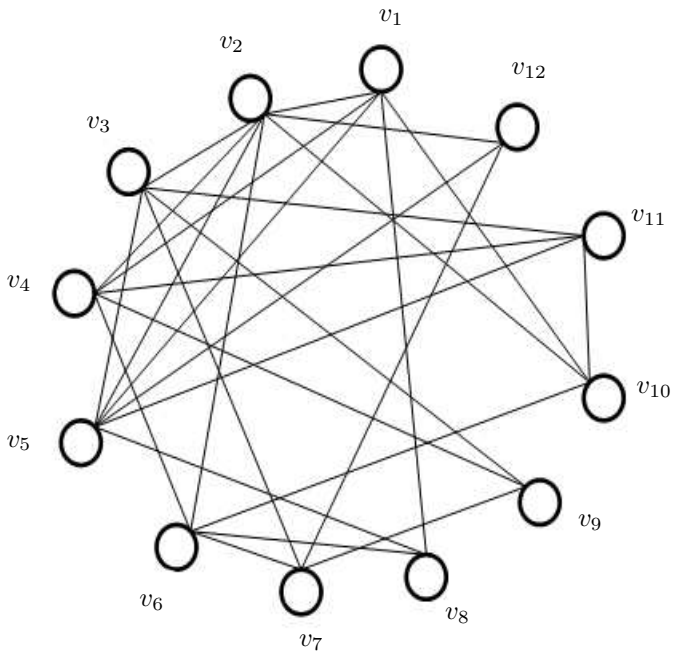
Berikut adalah daftar zat-zat kimia yang tidak dapat disimpan dalam satu ruangan. Seperti tampak pada Tabel 2.

Tabel 2: Konflik zat kimia

Zat Kimia	Tidak Dapat Disimpan Bersama
A	B, D, E, H, J
B	A, C , D, E, F, J, L
C	B, E, G, I, K
D	A, B, F, I, K
E	A, B, C, H, K, L
F	B, D, G, H, J
G	B, F, I, L
H	A, E, F
I	C, D, G
J	A, B, F, K
K	C, D, E, J
L	B, E, G

Kemudian, misalkan zat-zat kimia tersebut sebagai simpul yaitu A = v_1, B = v_2, ..., L = v_{12}.

Selanjutnya, modelkan permasalahan ini kedalam graf seperti tampak pada Gambar 2.



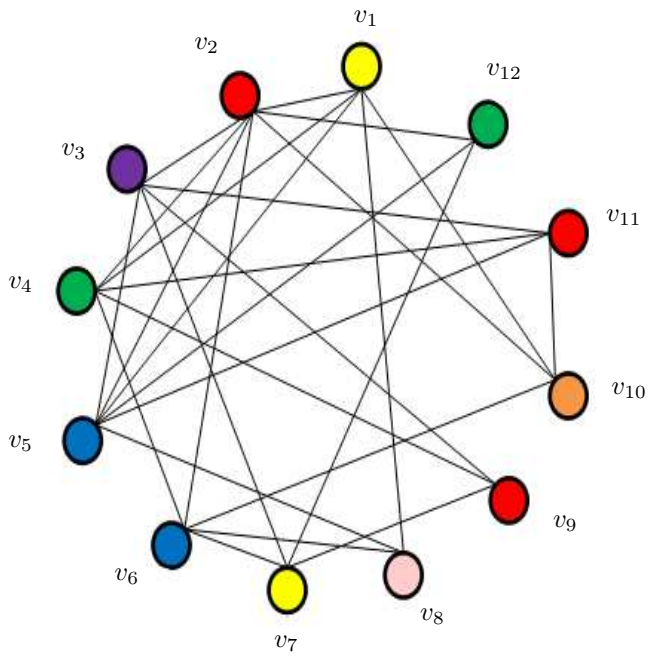
Gambar 2: Pemodelan dari konflik zat kimia ke dalam graf

Pada Gambar 2, zat-zat kimia dimisalkan sebagai simpul dan dimodelkan ke dalam graf yang terdiri dari 12 simpul $V = \{v_1, v_2, ..., v_{12}\}$.

Selanjutnya, dilakukan pewarnaan pada setiap simpul berdasarkan masing-masing algoritma untuk mendapatkan bilangan kromatik ($\chi(G)$) yang merupakan warna minimum dari suatu graf. Simpul dengan warna yang sama akan ditempatkan pada ruangan yang sama.

1. Penyelesaian Menggunakan Algoritma *D'satur*

   Setelah dilakukan pewarnaan berdasarkan langkah-langkah yang telah diuraikan pada Bagian 4, diperoleh graf seperti tampak pada Gambar 3.



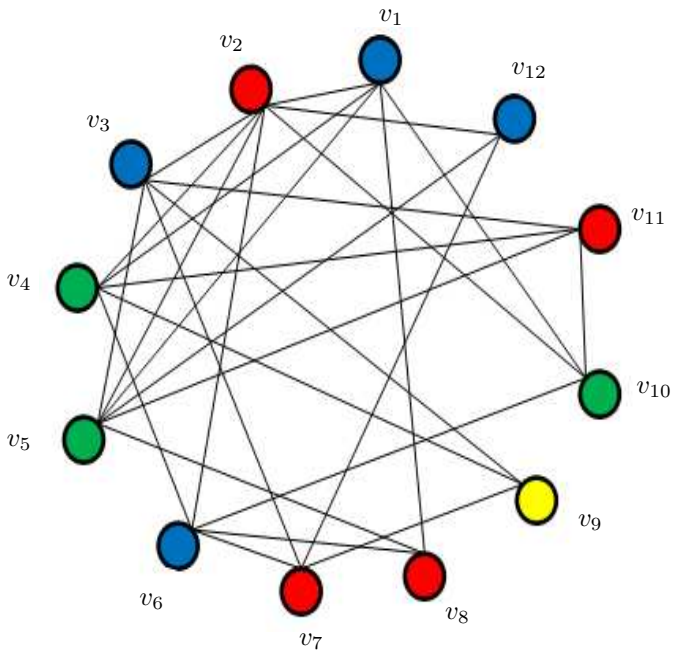
Gambar 3: Pewarnaan simpul menggunakan algoritma *d'satur*

   Pada Gambar 3, pewarnaan dengan menggunakan algoritma *d'satur* menghasilkan warna minimum $\chi(G) = 7$. Jadi, ruangan yang dibutuhkan untuk menyimpan zat-zat kimia tersebut dengan menggunakan algoritma *d'satur* adalah 7 ruangan. Simpul v_1, v_7 ditempatkan pada ruangan 1, v_2, v_9, v_{11} ditempatkan pada ruangan 2, v_3 ditempatkan pada ruangan 3, v_4, v_{12} ditempatkan pada ruangan 4, v_5, v_6 ditempatkan pada ruangan 5, v_8 ditempatkan pada ruangan 6, dan v_{10} ditempatkan pada ruangan 7.

2. Penyelesaian Menggunakan Algoritma *Vertex Merge*

   Setelah dilakukan pewarnaan berdasarkan langkah-langkah yang diuraikan pada Bagian 5, diperoleh graf seperti tampak pada Gambar 4.

Pada Gambar 4, pewarnaan dengan menggunakan algoritma *vertex merge* menghasilkan warna minimum $\chi(G) = 4$. Jadi, ruangan yang dibutuhkan untuk menyimpan zat-zat kimia tersebut adalah 4 ruangan. Simpul v_1, v_3, v_6, v_{12}



Gambar 4: Pewarnaan simpul menggunakan algoritma *vertex merge*

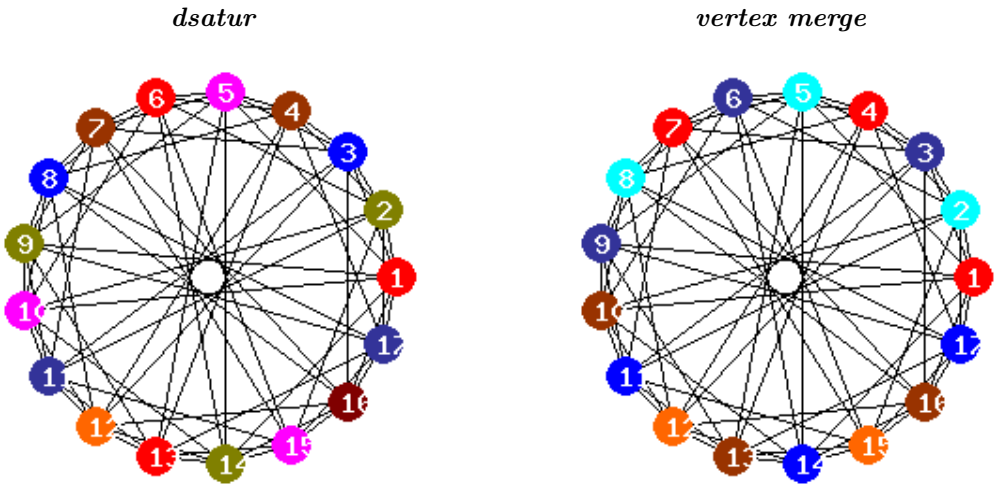
ditempatkan pada ruangan 1, v_2, v_7, v_8, v_{11} ditempatkan pada ruangan 2, v_4, v_5, v_{10} ditempatkan pada ruangan 3, dan v_9 ditempatkan pada ruangan 4.

Dari contoh permasalahan ruang penyimpanan zat-zat kimia di atas dapat di lihat perbandingan dari kedua algoritma. Penyelesaian menggunakan algoritma *d'satur* dibutuhkan 7 ruangan untuk penyimpanan zat-zat kimia tersebut. Sedangkan untuk algoritma *vertex merge*, cukup dibutuhkan 4 ruang penyimpanan saja. Jadi, penggunaan algoritma *vertex merge* lebih efisien dibandingkan algoritma *d'satur* karena biaya yang dibutuhkan untuk membangun ruang penyimpanan dapat diminimalisir.

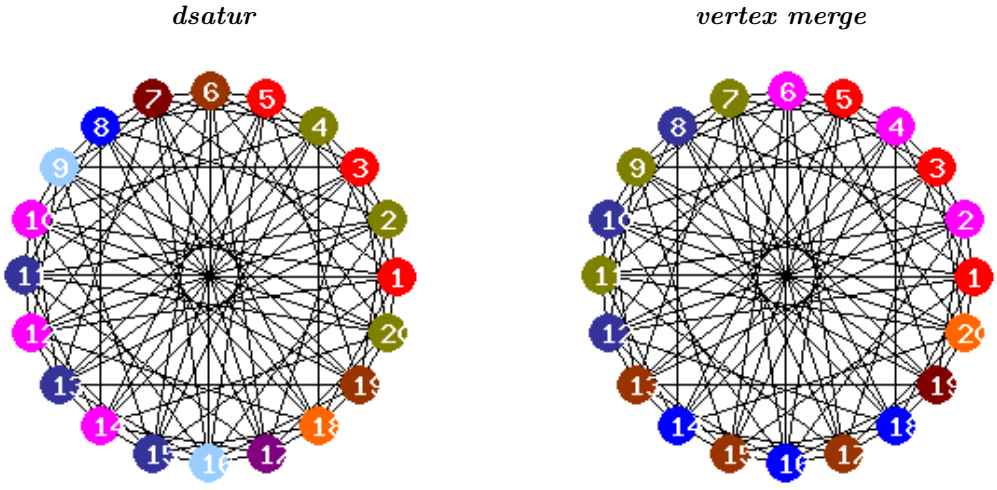
Untuk persoalan pewarnaan simpul graf yang lebih kompleks dimana terdapat simpul dan sisi dalam jumlah besar yang mungkin sulit diselesaikan secara manual, selain terdapat tingkat kesulitan yang lebih tinggi, juga akan memakan waktu yang lama. Oleh karena itu dianjurkan menggunakan pemrograman komputer. Pemrograman yang digunakan dalam artikel ini adalah pemrograman berbasis *php* dengan memakai sistem operasi *Linux*.

Berikut diberikan 3 buah contoh perbandingan algoritma *d'satur* dan algoritma *vertex merge* dengan menggunakan pemrograman *php* seperti tampak pada Gambar 5, 6 dan 7.

Pada Gambar 5, graf G_A memiliki 17 simpul yaitu $\{v_1, v_2, ..., v_{17}\}$. Pewarnaan simpul dengan menggunakan algoritma *d'satur* menghasilkan sebanyak 8 warna $(\chi(G_A)) = 8$. Sedangkan dengan menggunakan algoritma *vertex merge* menghasilkan 6 warna $(\chi(G_A)) = 6$.

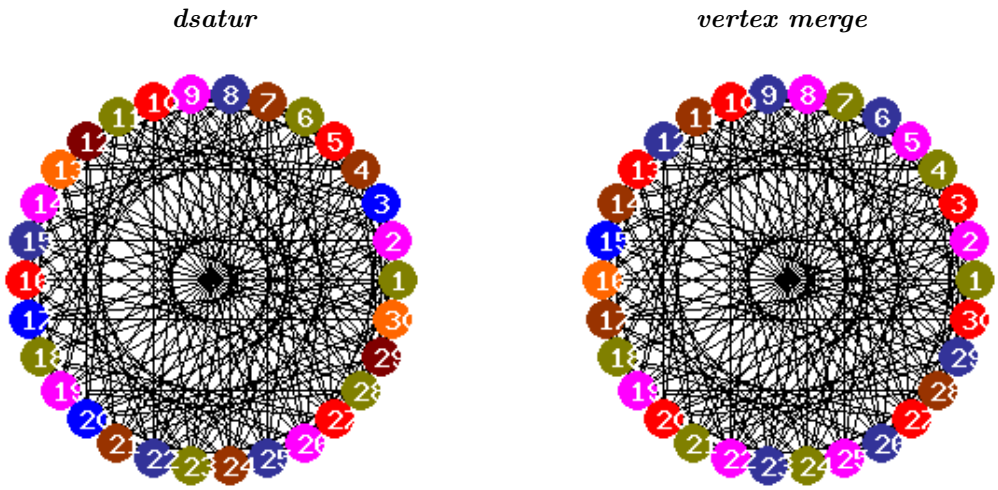


Gambar 5: Perbandingan graf G_A menggunakan pemrograman *php*



Gambar 6: Perbandingan graf G_B menggunakan pemrograman *php*

Pada Gambar 6, graf G_B memiliki 20 simpul yaitu $\{v_1, v_2, ..., v_{20}\}$. Pewarnaan simpul dengan menggunakan algoritma *d'satur* menghasikan 10 warna $(\chi(G_B)) = 10$. Sedangkan dengan menggunakan algoritma *vertex merge* menghasilkan 8 warna $(\chi(G_B)) = 8$.



Gambar 7: Perbandingan graf G_C menggunakan pemrograman *php*

Pada Gambar 7, graf G_C memiliki 30 simpul yaitu $\{v_1, v_2, ..., v_{30}\}$. pewarnaan simpul dengan menggunakan algoritma *d'satur* menghasilkan 8 warna $(\chi(G_C)) = 8$. Sedangkan dengan menggunakan algoritma *vertex merge* menghasilkan 6 warna $(\chi(G_C)) = 6$.

Berikut diberikan tabel perbandingan jumlah warna simpul antara algoritma *d'satur* dan algoritma *vertex merge* seperti tampak pada Tabel 3.

Tabel 3: Tabel perbandingan jumlah warna simpul

Jumlah Simpul	Jumlah Warna	
	D'satur	*Vertex Merge*
10	6	5
17	8	6
20	10	8
25	9	7
30	8	6

Pada Tabel 3, Graf dengan 10 simpul jika diselesaikan dengan algoritma *d'satur* menghasilkan 6 warna, sedangkan dengan algoritma *vertex merge* menghasilkan 5 warna. Selanjutnya, graf dengan 17 simpul jika diselesaikan dengan algoritma *d'satur* menghasilkan 8 warna, sedangkan dengan algoritma *vertex merge* menghasilkan 6 warna. Seterusnya, graf dengan 30 simpul jika diselesaikan dengan

algoritma *d'satur* menghasilkan 8 warna, sedangkan dengan algoritma *vertex merge* menghasilkan 6 warna.

Dari Tabel 3 terlihat jelas bahwa algoritma *vertex merge* selalu menghasilkan bilangangan kromatik (χ(G)) yang lebih kecil atau menghasilkan warna yang lebih sedikit dari pada algoritma *d'satur*. Artinya, algoritma *vertex merge* lebih baik digunakan dari pada algoritma *d'satur* dalam pewarnaan simpul graf tak berarah.

Secara umum kedua algoritma yang dibahas bisa digunakan dalam pewarnaan simpul graf tak berarah. Akan tetapi algoritma *vertex merge* lebih unggul dari pada algoritma *d'satur* karena selalu memiliki bilangan kromatik ($\chi(G)$) yang lebih kecil atau menghasilkan warna yang lebih sedikit.

DAFTAR PUSTAKA

[1] Brelaz, D. 1979. New Methods to Color the Vertices of a Graph. *Communication of the ACM*. 22: 251–256.

[2] Chartrand, G & L. Lesniak. 1986. *Graph and Digraph, 3^{rd} Edition*. Wadsworth, Inc., California.

[3] Deo, N. 1974. *Graph Theory with Application to Engineering and Computer Science*. Prentice-Hall International, Inc., London.

[4] Gros, J. L & J. Yellen. 2006. *Graph Theory and its Applications, 2^{nd} edition*. Chapman and Hall/ CRC. New York.

[5] Handrizal, A. Noraziah & A. N Abdalla. 2011. Comparation between Vertex Merge Algorithm and D'satur Algorithm. *Journal of Computer Science*. 7(5): 664–670.