Pencarian *Clique* Maksimal dan Bilangan *Clique* pada Graf Sederhana Menggunakan Modifikasi Algoritma *Clique*

Shintia Pratiwi[#1], Armiati[*2], Dewi Murni[*3]

[#]*Student of Mathematics Department Universitas Negeri Padang, Indonesia*
[*]*Lecturers of Mathematics Department Universitas Negeri Padang, Indonesia*

[1]shintiapratiwi579@gmail.com
[2]armiati_math_unp@yahoo.co.id
[3]dewimunp@gmail.com

Abstract – Finding maximal clique is a problem to get a clique with the maximum number of points on a graph, where the selected points are the points that are connected to each other. Finding maximal clique in a graph would be easier if using an algorithm. One of algorithms that can be used to determine the maximal clique is clique algorithm. However, the clique algorithm still has a weakness that the process is a relatively long process because it has two procedures. Therefore, in this study a modification to this algorithm. An algorithm not only must be correct, but also must be efficient. The efficiency of an algorithm is measured from the execution time of the algorithm and the space of memory that is required to run it. After modification is done to clique algoritma, the time of complexity asymptotic algorithm gained is $T(n) = O(n^3)$.

Keywords – graph, adjacency matriks, algorithm complexity, clique algorithm, maximal clique and clique number

Abstract – Pencarian *clique* maksimal merupakan suatu permasalahan untuk mendapatkan *clique* dengan jumlah titik terbanyak (maksimal) pada sebuah graf, dimana setiap titik yang terpilih merupakan titik yang saling terhubung satu sama lain. Pencarian *clique* maksimal pada suatu graf akan lebih mudah jika menggunakan algoritma. Salah satu algoritma yang dapat digunakan untuk menentukan *clique* maksimal adalah algoritma *clique*. Namun, algoritma *clique* masih memiliki kelemahan yaitu pada proses pengerjaannya yang relatif lama karena memiliki dua prosedur. Oleh karena itu, dilakukan modifikasi pada algoritma ini. Sebuah algoritma tidak saja harus benar, tetapi juga harus mangkus (*efisien*). Kemangkusan algoritma diukur dari waktu (*time*) eksekusi algoritma dan ruang (*space*) memori yang dibutuhkan untuk menjalankannya. Setelah dilakukan modifikasi pada algoritma *clique* diperoleh kompleksitas waktu asimptotik algoritma adalah $T(n) = O(n^3)$.

Kata Kunci – graf, matriks ketetanggaan graf, komplesitas algoritma, algoritma *clique*, *clique* maksimal dan bilangan *clique*.

PENDAHULUAN

Clique merupakan sebuah subgraf lengkap tak berarah dari suatu graf yang terdiri atas tiga titik atau lebih. Graf lengkap merupakan graf yang setiap dua titik berbeda pada graf tersebut dihubungkan oleh sebuah sisi. Bilangan *clique* merupakan order dari *clique* maksimal yang dinotasikan dengan $\omega(G)$.

Pencarian *clique* maksimal merupakan suatu permasalahan untuk mendapatkan *clique* dengan jumlah titik terbanyak (maksimal) dalam suatu graf. Permasalahan ini termasuk kedalam permasalahan *Non-Deterministik Polinomial* (*NP-Complete*), dimana untuk mendapatkan penyelesaian yang eksak dan efisien dari permasalahan tersebut sangat susah. Suatu permasalahan dikatakan *NP-Complete* apabila permasalahan tersebut termasuk dalam permasalahan *intractable* yang telah berhasil dibuktikan. Permasalahan *intractable* merupakan permasalahan yang belum ada algoritma yang efisien untuk memecahkan permasalahan tersebut.

Aplikasi *Clique* maksimal banyak diterapkan dalam bidang keilmuan seperti teknik, bisnis, ilmu kedokteran, kimia dan lainnya. Pada bidang teknik, *clique* maksimal diterapkan pada permasalahan lampu lalu lintas, yaitu untuk mengoptimalkan siklus waktu lampu lalu lintas pada persimpangan dengan memperhatikan setiap arah lalu lintas yang kompatibel. Dua arah lalu lintas dikatakan kompatibel jika kedua arah tersebut dapat berjalan secara bersamaan. Pada permasalahan ini, arah lalu lintas dilambangkan dengan titik dan arah lalu lintas yang kompatibel dilambangkan dengan sisi.

Pencarian *clique* maksimal dan bilangan *clique* pada graf yang memiliki titik dan sisi banyak secara manual sangat sulit, sehingga diperlukan suatu algoritma untuk menyelesaikannya. Algortima merupakan urutan logis langkah-langkah penyelesaian masalah secara sistematis [3].

Terdapat beberapa algoritma pencarian yang dapat digunakan untuk menentukan *clique* maksimal suatu graf, diantaranya Algoritma *Brute Force* dan Algoritma *Back-Tracking*. Jika dilihat dari kemangkusan kedua algoritma ini, Algoritma *Back-Tracking* memberikan hasil yang cukup baik dari pada Algoritma *Brute Force*. Akan tetapi Algoritma *Back-Traacking* masih memiliki kompleksitas waktu terburuk yaitu $O(n^k)$ [2].

Salah satu algoritma yang juga dapat digunakan untuk menentukan *clique* maksimal dan bilangan *clique* adalah algoritma *clique*. Akan tetapi, algoritma ini masih memiliki kelemahan pada proses pengerjaannya. Pencarian *clique* pada algoritma ini dilakukan untuk ukuran *clique* yang diinginkan dan akan berhenti apabila telah ditemukan *clique* dengan ukuran tersebut. Akan tetapi, ukuran *clique* yang diinginkan ini belum tentu merupakan *clique* maksimal. Sehingga, pada penelitian ini dilakukan modifikasi pada algoritma *clique* agar dapat menentukan *clique* dengan ukuran maksimal.

Suatu algoritma tidak saja harus benar, akan tetapi juga harus mangkus (*efisien*). Kemangkusan algoritma diukur dari waktu (*time*) eksekusi algoritma dan ruang (*space*) memori yang dibutuhkan untuk menjalankannya. Oleh karena itu, pada penelitian ini akan ditentukan kompleksitas waktu dari modifikasi algoritma *clique* untuk menentukan *clique* maksimal dan bilangan *clique* pada suatu graf. Selanjutnya, modifikasi algoritma *clique* akan diterjemahkan kedalam bahasa pemograman Matlab secara bertahap, dan dilakukan uji coba terhadap program yang dibuat yaitu untuk menentukan *clique* maksimal dan bilangan *clique* pada graf kompatibel persimpangan empat, lima, dan enam.

METODE

Penelitian ini merupakan penelitian dasar (teoritis). Metode yang digunakan adalah metode deskriptif dengan menganalisa teori-teori yang relevan dengan permasalahan yang dibahas dan berlandaskan pada studi kepustakaan. Adapun langkah-langkah yang dilakukan untuk menjawab permasalahan adalah sebagai berikut:

1. Membuat tahapan modifikasi algoritma *clique* untuk menentukan *clique* maksimal dan bilangan *clique* pada graf sederhana.
2. Menentukan kompleksitas waktu asimptotik dari modifikasi algoritma *clique*.
3. Menerapkan modifikasi algoritma dalam bentuk program komputer yaitu pada *software* Matlab.
4. Melakukan uji coba terhadap program yang telah dibuat dengan menerapkannya pada graf kompatibel di persimpangan empat, lima dan enam.
5. Menarik kesimpulan dari hasil dan pembahasan yang telah dikerjakan.

HASIL DAN PEMBAHASAN

A. *Modifikasi Algoritma Clique*

Modifikasi yang dilakukan pada algoritma *clique* yaitu pada proses pencarian *clique* tanpa menentukan ukuran *clique* yang akan dicari. Hal ini dilakukan karena, ukuran *clique* yang akan dicari belum tentu merupakan *clique* maksimal. Pencarian *clique* maksimal pada modifikasi algoritma *clique* dimulai dengan menentukan titik yang memiliki derajat maksimum, dan kemudian menentukan titik yang saling terhubung dengan titik yang memiliki derajat maksimum. Pencarian *clique* akan berhenti apabila tidak ada lagi titik yang saling terhubung dengan titik yang memiliki derajat maksimum selanjutnya.

B. *Kompleksitas Waktu Asimptotik Modifikasi Algoritma Clique*

Untuk menentukan komplesitas waktu asimptotik suatu algoritma terlebih dahulu harus ditentukan kompleksitas waktu algoritma tersebut. Misalkan banyaknya titik pada graf adalah n titik, maka kompleksitas waktu algoritma *clique* dapat ditentukan menggunakan langkah-langkah berikut:

1. Langkah pertama adalah menentukan titik yang memiliki derajat maksimum dengan cara memeriksa derajat setiap titik pada graf. Banyak langkah yang dibutuhkan untuk memeriksa derajat setiap titik pada graf yang memiliki n titik adalah n^2 langkah. Karena graf merupakan graf sederhana dengan n titik, maka graf tidak memiliki *loop*, sehingga hubungan titik dengan dirinya sendiri tidak perlu diperiksa. Jadi, jumlah langkah yang diperlukan untuk menentukan titik yang memiliki derajat maksimum pertama adalah $n^2 - n$ langkah.
2. Langkah kedua adalah menentukan semua titik yang saling terhubung dengan titik yang memiliki derajat maksimum. Untuk menentukannya dilakukan pencarian terhadap setiap titik pada graf. Karena graf memiliki n titik, maka banyak langkah untuk menentukan titik yang saling terhubung dengan titik yang memiliki derajat maksimum pertama adalah $n - 1$ langkah.
3. Langkah tiga adalah menentukan titik yang memiliki derajat maksimum kedua. Karena satu titik yang berderajat maksimum telah terpilih, sehingga jumlah titik pada graf menjadi $(n - 1)$ titik. Dengan cara yang sama seperti pada langkah 1, maka langkah yang dibutuhkan untuk menentukan titik yang berderajat maksimum selanjutnya adalah $((n - 1)^2 - (n - 1))$ langkah.

   Sehingga, banyak langkah yang diperlukan untuk menentukan semua titik yang berderajat maksimum adalah:

$$\{(n-2)^2-(n-2)\}+\{(n-3)^2-(n-3)\}$$
$$+\{(n-4)^2-(n-4)\}$$
$$+\{(n-5)^2-(n-5)\}+\cdots$$
$$+\left\{\left(n-(n-2)\right)^2-\left(n-(n-2)\right)\right\}$$
$$+\left\{\left(n-(n-1)\right)^2-\left(n-(n-1)\right)\right\}$$

4. Langkah 4 adalah menentukan titik yang saling terhubung dengan titik yang memiliki derajat maksimum kedua. Untuk menentukannya, langkah yang dilakukan sama dengan langkah 2. Sehingga, banyak pencarian yang dilakukan untuk menemukan titik yang saling terhubung dengan titik berderajat maksimum kedua adalah $((n-1)-1)=(n-2)$ langkah.
   Sehingga, banyak langkah yang diperlukan untuk menentukan semua titik yang saling terhubung dengan titik berderajat maksimum adalah:

$$(n-2)+(n-3)+(n-4)+\cdots+\left(n-(n-4)\right)$$
$$+\left(n-(n-3)\right)+\left(n-(n-4)\right)$$
$$+\left(n-(n-3)\right)+\left(n-(n-2)\right)$$
$$+\left(n-(n-1)\right)$$

5. Jadi, jumlah semua langkah yang diperlukan algoritma *clique* untu menentukan *clique* maksimal pada suatu graf sederhana adalah:

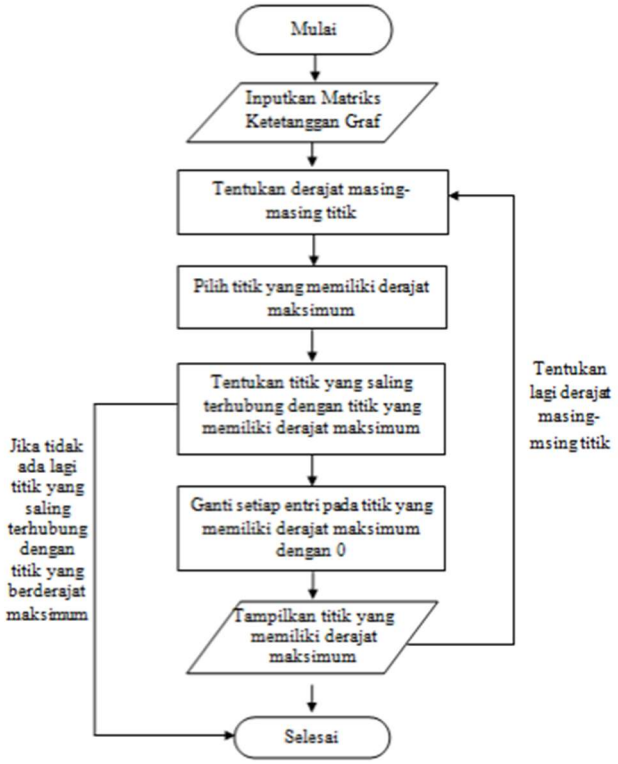
$$\Big[\{(n-2)^2-(n-2)\}+\{(n-3)^2-(n-3)\}+$$
$$\{(n-4)^2-(n-4)\}+\{(n-5)^2-$$
$$(n-5)\}+\cdots+\left\{\left(n-(n-5)\right)^2-\right.$$
$$\left.\left(n-(n-5)\right)\right\}+\left\{\left(n-(n-4)\right)^2-\right.$$
$$\left.\left(n-(n-4)\right)\right\}+\left\{\left(n-(n-3)\right)^2-\right.$$
$$\left.\left(n-(n-3)\right)\right\}+\left\{\left(n-(n-2)\right)^2-\right.$$
$$\left.\left(n-(n-2)\right)\right\}+\left\{\left(n-(n-1)\right)^2-\right.$$
$$\left.\left(n-(n-1)\right)\right\}+\left\{\left(n-(n-0)\right)^2-\right.$$
$$\left.\left(n-(n-0)\right)\right\}\Big]+\big[(n-2)+$$
$$(n-3)+(n-4)+\cdots+\left(n-\right.$$
$$\left.(n-4)\right)+\left(n-(n-3)\right)+$$
$$\left(n-(n-4)\right)+\left(n-(n-3)\right)+$$
$$\left(n-(n-2)\right)+\left(n-(n-1)\right)\big]$$
$$\ldots(1)$$

   Dengan menyelesaikan persamaan (1) secara aljabar, diperoleh kompleksitas waktu algoritma *clique* adalah $\frac{2n^3+3n^2-5n}{6}$, dan kompleksitas waktu asimptotik algoritma *clique* adalah n^3.

C. *Desain Proses Program Modifikasi Algoritma Clique untuk menentukan Clique Maksimal*

Pada proses ini, modifikasi algoritma *clique* akan diterjemahkan ke bahasa pemrograman Matlab secara bertahap. Dalam pembuatan program aplikasi, graf yang akan ditentukan *clique* maksimal dan bilangan *clique* nya direpresentasikan dalam bentuk matriks ketetanggaan. Kemudian, ditentukan derajat dari masing-masing titik pada graf, selanjutnya dilakukan pencarian untuk menentukan titik yang memiliki derajat maksimum. Langkah selanjutnya adalah menentukan satiap titik yang terhubung dengan titik yang memiliki derajat maksimum. Pencarian dilakukan sampai tidak ada lagi titik yang saling terhubung dengan titik yang memiliki derajat maksimum.

Berikut dijelaskan desain proses dari algoritma *clique*.

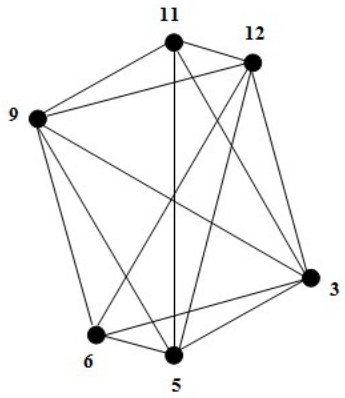


Gambar 1. *Flow Chart* Proses Pembentukan Modifikasi Algoritma *Clique*

D. *Hasil Program Algoritma Clique untuk Menentukan Clique Maksimal dan Bilangan Clique pada Graf Kompatibel di Persimpangan Empat, Lima, dan Enam*

1. Hasil Program Algoritma *Clique* pada Graf Kompatibel di Persimpangan Empat

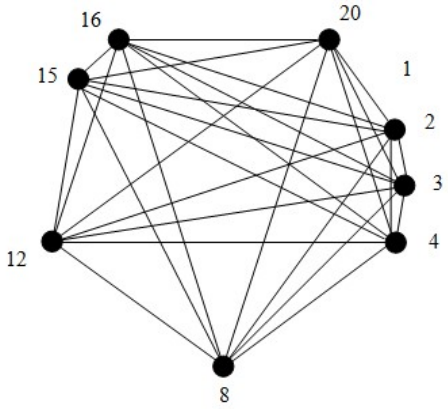
Dari program Matlab diperoleh *clique* maksimal dari graf kompatibel pada persimpangan empat yaitu {3, 6, 9, 12, 11, 5}, dan bilangan *clique* nya adalah 6. Kompleksitas waktu yang dibutuhkan algoritma *clique* untuk menentukan *clique* maksimal dan bilangan *clique* pada graf kompatibel persimpangan empat adalah $\frac{2n^3+3n^2-5n}{6}=\frac{2(12^3)+3(12^2)-5(12)}{6}=\frac{3.456+432-6}{6}=$ 648 satuan waktu. Artinya, misalkan bahwa satu langkah pada algoritma *clique* membutuhkan waktu 2 detik, maka untuk n = 12 kebutuhan waktu algoritma untuk menentukan *clique* maksimal adalah 1.276 detik atau 21 menit 16 detik. *Clique* maksimal dari graf kompatibel pada persimpangan empat adalah sebagai berikut:



Gambar 2. *Clique* Maksimal Graf Kompatibel pada Persimpangan Empat

2. Hasil Program Algoritma *Clique* pada Graf Kompatibel di Persimpangan Lima

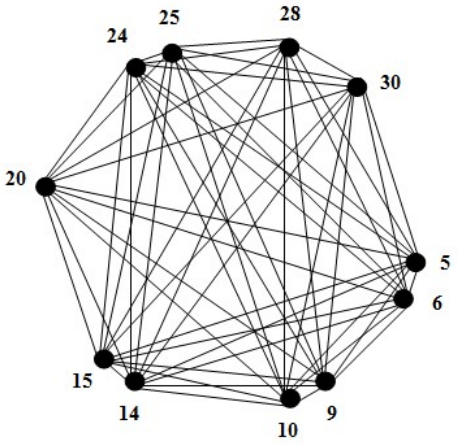
Dari program Matlab diperoleh *clique* maksimal dari graf kompatibel pada persimpangan lima yaitu {4, 8, 12, 16, 20, 3, 15, 2}, dan bilangan *clique* nya adalah 8. Kompleksitas waktu yang dibutuhkan algoritma *clique* untuk menentukan *clique* maksimal dan bilangan *clique* pada graf kompatibel persimpangan lima adalah $\frac{2n^3+3n^2-5n}{6} = \frac{2(20^3)+3(20^2)-5(20)}{6} = \frac{16.000+\ .200-100}{6} =$ 2.850 satuan waktu. Artinya, misalkan bahwa satu langkah pada algoritma *clique* membutuhkan waktu 2 detik, maka untuk n = 20 kebutuhan waktu algoritma untuk menentukan *clique* maksimal adalah 2.850 detik atau 47 menit 30 detik. *Clique* maksimal dari graf kompatibel pada persimpangan empat adalah sebagai berikut:



Gambar 3. *Clique* Maksimal Graf Kompatibel pada Persimpangan Lima

3. Hasil Program Algoritma *Clique* pada Graf Kompatibel di Persimpangan Enam

Dari program Matlab diperoleh *clique* maksimal dari graf kompatibel pada persimpangan lima yaitu {5, 10, 15, 20, 25, 30, 9, 14, 24, 8, 6},dan bilangan *clique* nya adalah 11. Kompleksitas waktu yang dibutuhkan algoritma *clique* untuk menentukan *clique* maksimal dan bilangan *clique* pada graf kompatibel persimpangan enam adalah $\frac{2n^3+3n^2-5n}{6} = \frac{2(30^3)+3(30^2)-5(30)}{6} = \frac{54.000+\ .700-150}{6} =$ 9.425 satuan waktu. Artinya, misalkan bahwa satu langkah pada algoritma *clique* membutuhkan waktu 2 detik, maka untuk n = 20 kebutuhan waktu algoritma untuk menentukan *clique* maksimal adalah 18.850 detik atau 5 jam 14 menit 10 detik.*Clique* maksimal dari graf kompatibel pada persimpangan empat adalah sebagai berikut:



Gambar 3. *Clique* Maksimal Graf Kompatibel pada Persimpangan Enam

SIMPULAN

Kompleksitas waktu asimptotik algoritma *clique* adalah $T(n) = O(n^3)$. Notasi O-Besar ini melambangkan berapa waktu yang dibutuhkan algoritma untuk menyelesaikan masalah dengan meningkatnya ukuran *n*. Karena kompleksitas waktu asimptotik algoritma *clique* pada graf yang memiliki *n* titik adalah $O(n^3)$, maka algoritma ini termasuk kedalam algoritma kubik. Artinya, jika *n* = 100, maka kompleksitas waktu asimptotik algoritma adalah 1.000.000 satuan waktu.

Algoritma *clique* memiliki beberapa langkah untuk menentukan *clique* maksimal dan bilangan *clique* suatu graf pada *software* Matlab. Langkah pertama yaitu menentukan matriks ketetanggaan dari graf, selanjutnya menentukan titik yang memiliki derajat maksimum, dan kemudian menentukan titik-titik yang saling terhubung dengan titik yang memiliki derajat maksimum tersebut. Pencarian *clique* maksimal selesai saat tidak ada lagi titik yang saling terhubung dengan titik yang memiliki derajat maksimum.

Penerapan program aplikasi algoritma *clique* untuk menentukan *clique* maksimal dan bilangan *clique* menggunakan *software* Matlab dapat menemukan *clique* maksimal dan bilangan *clique* untuk setiap graf. Pada contoh penerapan yaitu graf kompatibel untuk persimpangan empat, lima, dan enam, program dapat menemukan *clique* maksimal dan bilangan *clique* dengan cepat. Sehingga, berdasarkan uji coba dan evaluasi pada contoh tersebut diperoleh kesimpulan bahwa program dapat berjalan dengan baik pada graf yang memiliki banyak titik.

REFERENSI

[1] Budayasa, Ketut. 2007. *Teori Graph dan Aplikasinya*. Surabaya: Universitas Negeri Surabaya.

[2] Lumbantobing, Adventus Wijaya. 2007. *Pencarian Clique Dalam Graf Dengan Menggunakan Algoritma Backtracking.* Jurnal: ITB. Vol. III. No. 2

[3] Munir, Rinaldi. 2012. *Matematika Diskrit*. Bandung: Informatika Bandung.

[4] Pratiwi, Shintia. 2016. *Pencarian Clique Maksimal dan Bilangan Clique Menggunakan Modifikasi Algoritma Clique*. Padang: Universitas Negeri Padang.