Sets Approach Algorithms for Determination of Component Minimal Complete Subgraph

Silviu Bârzã
Faculty of Mathematics and Informatics
Spiru Haret University
Bucharest, Romania
silviu.barza@gmail.com

Abstract
Giving a graph G we wish to determine the component minimal complete subgraph of G. For this goal we present two algorithms in terms of sets representation of graph G. We give too the theorems that characterized this algorithms.
Keywords: *subgraphs, complete graphs*
ACM/AMS Classification: 05C30

1. General definitions and results

Firstly I wish to give some of the graph theory definitions and a few results related to component minimal complete subgraphs as presented in [Bârzã, 2012].

Definition 1.1 *Let*

$$V = \{x_1, x_2, \ldots, x_n\}$$

be a finite and non-empty set and

$$E = \{\{x, y\} \,| x, y \in X, x \neq y\}.$$

*The pair $G = (V, E)$ is a **graph**, elements in V are named **vertices** and element in E are named **edge**.*

Definition 1.2 *Let $G = (V, E)$ be a graph. A sequence of vertices $y_1, y_2, \ldots, y_k$ is named **walk** in G if $\{y_i, y_{i+1}\} \in E$ for any $i = 1, 2, \ldots, k-1$. If $y_1 = y_k$ the walk is named **loop**. The walk (and the loop) is specified as*

$$L = [y_1, y_2, \ldots, y_k].$$

Definition 1.3 *Let $G = (V, E)$ be a graph. If $W \subset V$ and*

$$F = \{\{x, y\} \in E | x, y \in W\}$$

*then the graph $H = (W, F)$ is named **subgraph** of G.*

Definition 1.4 *Let $G = (V, E)$ be a graph. If for any $x, y \in V$ there exist a walk from x to y then G is named* ***connected****. If G is not connected, G is named* ***disconnected****.*

Definition 1.5 *Let $G = (V, E)$ be a graph. A subgraph $H = (W, F)$ of G which is connected and there does not exists a walk from x to y for any $x \in W$ and $y \in V - W$ is named* ***component*** *of G.*

Propozition 1.1 *Let $G = (V, E)$ be a graph. There exist a partition of V with the sets $V_1, V_2, \ldots, V_k$ ($V_1 \cup V_2 \cup \ldots \cup V_k = V$ and $V_i \cap V_j = \oslash$ for any $i, j = 1, 2, \ldots, k$, $i \neq j$) so that subgraphs $G_i = (V_i, F_i)$ are components in G.*

Definition 1.6 *Let $G = (V, E)$ be a graph with $|V| = p \leq 3$. If for any $x, y \in V$, $\{x, y\} \in E$ then G is named* ***complete*** *and we write it as K_p.*

Definition 1.7 *Let $G = (V, E)$ be a graph. If for any $x \in V$ there exist $y, z \in V$, $y \neq z$ so that $\{\{x, y\}, \{x, z\}, \{y, z\}\} \subset E$ we say that G is a* ***minimal complete graph****.*

Definition 1.8 *Let $G = (V, E)$ be a graph. If there exist $W \subseteq V$ so that the subgraph $H = (W, F)$ is a complete graph, with $|W| = p$ and for any $x \in V - W$ the subgraph $H' = (W - \{x\} .F')$ is not a complete graph, then G is named* ***K_p-maximal complete graph****.*

Definition 1.9 *Let $G = (V, E)$ be a graph. If any component in G is a minimal complete graph then G is named* ***component minimal complete graph****.*

Definition 1.10 *Let $G = (V, E)$ be a graph and $x \in V$. If we consider the set $C = \{y \in V | \{x, y\} \in E\}$ then the number $\omega(x) = |C|$ is named the* ***degree*** *of x.*

Let CMC_G designate the component minimal complete subgraph for the graph G, if such a subgraph exists.

Proposition 1.2. *Let $G = (V, E)$ be a graph and*

$$X = \{x \in V | \omega(x) = 0\}$$

with $V - X \neq \oslash$. Let consider the subgraph $H = (V - X, F)$ of G. Then, G has a component minimal complete subgraph if and only if H has a component minimal complete subgraph.
In addition, we have

$$CMC_G = CMC_H.$$

Proposition 1.3. *Let $G = (V, E)$ be a connected graph and $X = \{x \in V | \omega(x) = 1\}$. Let us consider the subgraph $H = (V - X, F)$ of G. Then, G gas a component minimal complete subgraph if and only if H has a component minimal complete subgraph and*

$$CMC_G = CMC_H.$$

Proposition 1.4. *Let $G = (V, E)$ be a graph and let $CMC_G = (U, F)$ be the component minimal complete subgraph for G. Then for any $x \in U$ in G we have $\omega(x) \geq 2$.*

2. **Algorithm without vertexes and edges elimination**

In this section we give an algorithm based only on definition 1.7.

Let $G = (V, E)$ be a grap $k = |E|$ and so, $E = \{m_1, m_2, \ldots, m_k\}$. We consider the following process:

Step 1. Let $U = \oslash$, $F = \oslash$ and $i = 1$.
Step 2. If $i = k-2$ then STOP otherwise continue.
Step 3. Let $j = i + 1$.
Step 4. If $j = k - 1$ then $i \leftarrow i + 1$ and go to Step 2, otherwise continue.
Step 5. If $m_i \cap m_j = \oslash$ then go to Step 9, otherwise continue.
Step 6. Let $p = (m_i \cup m_j) \backslash (m_i \cap m_j)$.
Step 7. If $p \notin \{m_{j+1}, \ldots, m_k\}$ then go to Step 9, otherwise continue.
Step 8. Let $U \leftarrow U \cup m_i \cup m_j$ and $F \leftarrow F \cup \{m_i, m_j, p\}$.
Step 9. Let $j \leftarrow j + 1$ and go to Step 4.

Now let us analyze this process. Because the process is based on variation of i from 1 to $k - 2$ and on variation of j from $i + 1$ to $k - 1$ it follows that described process is finite and it might be an algorithm to solve the problem of determination of component minimal complete subgraph. To show it, we have to determine the result obtains when process is finished.

In Step 8 we consider that U is replaced by $U \cup m_i \cup m_j$ with U included in E. If x belong to resulting set, then x belongs to old U or $x \in m_i \cup m_j$.

If x belongs to old U then we go back to previously modification of U. So it exits $t \in \{1, 2, \ldots, k\}$ so that $x \in m_t$ with $m_t = \{x, y\}$, $y \in V$. Because construction of U, it follows that it exists $s \in \{1, 2, \ldots, k\}$, $s \neq t$, so that in one step we had the substitution $U \leftarrow U \cup m_t \cup m_s$.

If $x \in m_i \cup m_j$ the we can consider $t = i$ and $s = j$.

The substitution $U \leftarrow U \cup m_t \cup m_s$ appears if and only if in Step 5 we have $m_t \cap m_s \neq \oslash$ and in Step 7 we have $(m_t \cup m_s) \backslash (m_t \cap m_s) = p \in E$. So either $x \in m_s$ or $x \in p$ and it follows that $(m_t \cup m_s, \{m_t, m_s, p\})$ is a complete graphs with 3 vertexes.

We obtain that for any $x \in U$, there exists $y, z \in U$ so that

$$(\{x, y, z\}, \{\{x, y\}, \{x, z\}, \{y, z\}\})$$

subgraph in (U, F), and so $(U, F) = CMC_G$. That means that when process described above is finished, it produces expected results.

Now let us determine the complexity for this algorithm. We will give the complexity in terms of testing operation.

In Step 7 the condition is specified as $p \in \{m_{j+1}, \ldots, m_k\}$. In the worse case, to test this condition p must be compared with all the values from the set $\{m_{j+1}, \ldots, m_k\}$ which has $m - j$ elements.

In Step 5 we make one more test and so, for a fixed value of j, we have a maximum number of tests equal with $k - j + 1$.

For a fixed value of i, the steps between 5 and 9 are repeated for any j from $i + 1$ to $k - 1$ and so the maximum number of test for a fixed value of i

is determined by:

$$\sum_{j=i+1}^{k-1} k+1-j = \frac{1}{2}(k^2+k-2) - \frac{1}{2}(2k+1)i + \frac{1}{2}i^2.$$

Because i is varying from 1 to $k-2$ it follows that the maximum number of test operation needed to finish the algorithm is determined by:

$$\sum_{i=1}^{k-2} \frac{1}{2}(k^2+k-2) - \frac{1}{2}(2k+3)i - \frac{1}{2}i^2 = \frac{1}{6}(k^3-9k^2-7k+6)$$

So, the maximum number of testing operation from the algorithm is $O(k^3)$

Now we can resume the above analysis and we can give the following result:

Theorem 2.1. *Let $G = (V, E)$ be a graph with $|E| = k$. Then by applying the algorithm given above we obtain CMC_G and the algorithm has the complexity $O(k^3)$.*

As we can see, in the above algorithm we use only the edges set and so the result specified in theorem 2.1. is not influenced by existence of isolated vertexes.

In the future we specify this algorithm as ***algorithm for determination of CMC_G without elimination***.

3. **Algorithm with vertexes and edges elimination**

In this section we wish to specify a process for determination of CMC_G which to be faster than the *algorithm for determination of CMC_G without elimination*. To speeds up the existing algorithm we have to consider the result indicated by proposition 1.2, and 1.3 specified in first section of this paper which are firstly given in [Bârzã, 2012].

Let us consider $H = (X, T)$ as original given graph.

By applying proposition 1.2, related to isolated vertex, we obtain a new graph in which the set of vertexes can be smaller the X and with the same CMC as the original graph, but how we see in the final of section 2, the elimination of isolated vertexes has no influence on complexity of the algorithm.

By applying proposition 1.3, related to vertexes with degree equal with 1, we obtain a new graph with the same CMC as original one, in which both the set of vertexes and the set of edges are smaller. We know already that a smaller set of vertexes has no influence on complexity of the algorithm, but a smaller set of edges eliminates testing operation and so, the application of *algorithm for determination of CMC without elimination* will be finished faster when it is applied to new graph.

One application of proposition 1.2 and 1.3 do not give us the guaranties the in new graph do not exist anymore the vertexes with degree equals with 0 or 1. This fact can be shown by the next cases.

Case 1. In graph G there exists a vertex x with $\omega(x) = 1$ which forms an edge $\{x, y\}$ and $\omega(y) = 2$. By applying proposition 1.3, the vertex x is eliminated and also it is eliminated the edge $\{x, y\}$, and so, in the new graph we have that $\omega(y) = 1$.

Case 2. We can consider that the original graph has a component formed with the set of vertexes $\{x, y, z\}$ and with the set of edges $\{\{x, y\}, \{y, z\}\}$. Because we have a component, we have $\omega(x) = 1$, $\omega(z) = 1$, and $\omega(y) = 2$. Now, applying proposition 1.3 we eliminate the vertexes x and z, and both edges which form the component and we obtain a new component formed only with y as an isolated vertex.

The cases presented indicate that the application of proposition 1.2 and 1.3 must be re-iterated until do not exist the vertexes with degree equal with 0 or 1.

If we consider that after all possible application of proposition 1.2 and 1.3 we obtain a new graph $G = (V, E)$, then for this graph it is true the result given in proposition 1.4. In the same time, in agreement with proposition 1.2 and 1.3, we have $CMC_H = CMC_G$.

If we consider the above considerations, we can give the following process:

Step 1. Let $Y = \{x \in X | \omega(x) = 0\} \cup \{x \in X | \omega(x) = 1\}$.

Step 2. If $Y = \oslash$ then go to Step 4, otherwise continue.

Step 3. Let $X \leftarrow X/Y$, $T \leftarrow T/\{m \in T | m \cap Y \neq \oslash\}$ and go to Step 1.

Step 4. Let $V = X$ $E = T$, $k = |E|$ and the graph $G = (V, E)$ for which we consider that $E = \{m_1, m_2, \ldots, m_k\}$.

Step 5. Let $U = \oslash$, $F = \oslash$ and $i = 1$.

Step 6. If $i = k{-}2$ then STOP otherwise continue.

Step 7. Let $j = i + 1$.

Step 8. If $j = k - 1$ then $i \leftarrow i + 1$ and go to Step 2, otherwise continue.

Step 9. If $m_i \cap m_j = \oslash$ then go to Step 9, otherwise continue.

Step 10. Let $p = (m_i \cup m_j) \backslash (m_i \cap m_j)$.

Step 11. If $p \notin \{m_{j+1}, \ldots, m_k\}$ then go to Step 9, otherwise continue.

Step 12. Let $U \leftarrow U \cup m_i \cup m_j$ and $F \leftarrow F \cup \{m_i, m_j, p\}$.

Step 13. Let $j \leftarrow j + 1$ and go to Step 4.

This process used inside the *algorithm for determination of CMC without elimination* (Steps from 5 to 13) and so the process is finite because the elimination part (Steps from 1 to 3) is clearly finite.

As we specified before process presentation, the elimination part (Steps from 1 to 3) generate a graph G (indicated in Step 4) so that $CMC_H = CMC_G$ and the *algorithm for determination of CMC without elimination* part (Steps from 5 to 13) effectively generate CMC_G.

In worse case for this process, from the elimination part it results that $H = G$ and so the maximum number o test operation is the same as in case of *algorithm for determination of CMC without elimination.*

The analysis from above can be resumed in the following result:

Theorem 3.1. *Let $H = (X, T)$ be a graph. Then by applying the algorithm given above we obtain CMC_H and the algorithm has the complexity $O(k^3)$.*

In the future we call this algorithm as ***algorithm for determination of*** CMC ***with elimination***.

We consider that is obvious that se second algorithm in faster because the same operations are applied to a smaller set if there exists removed edges and the two algorithms have the same speed if no edges are eliminated.

4. **Conclusions and supplementary results**

Obtaining component minimal complete subgraph can be interesting, but someone could be interested in effectively finding all subgraphs of a given graph that are complete graph with three vertexes.

The goal can be easily reached just with minor modification of the algorithms indication in section 2 and 3.

So we must add a new set, let say K_3 which will be initialized by $K_3 = \oslash$ in Step 1 of the *algorithm for determination of* CMC *without elimination* (or in Step 5 of the *algorithm for determination of* CMC *with elimination*).

The set K_3 will keep the sets of three vertexes which form complete graphs with three vertexes and are subgraphs in processed graph.

The second modification is to memorize a complete graph with three vertexes once the algorithms determine it and to do so we must add the replacement $K_3 \leftarrow K_3 \cup \{m_i \cup m_j\}$ to Step 8 of the *algorithm for determination of* CMC *without elimination* (or in Step 12 of the *algorithm for determination of* CMC *with elimination*).

If finding of all complete subgraphs with three vertexes is our goal, then we can rename the algorithms presented above replacing CMC from their name with CK_3, and so to speak about the *algorithm for determination of* CK_3 *without elimination* and the *algorithm for determination of* CK_3 *with elimination*).

We consider that it is interesting to study what it is happened if we can apply the algorithms to disconnected graphs so that the determination of CMC works separately on every component of a given graph. I wish to do this task in a future paper. Also I wish to give in a next paper similar algorithm for algebraic approach.

As it is well known, K_3 belongs to so called *perfect graphs.* So it can be interesting if similar algorithms can be found for other type of perfect graphs.

References

1. Bamdy, J.A., Murty U.S.R., *Graph Theory*, "Springer-Verlag", 2007.
2. Bârzã, S., and Morogan, L.M., *Algoritmica grafurilor*, "Editura Fundatiei Romania de Maine", Bucharest, 2008.
3. Bârzã, S., *Some Consideration about Component Minimal Complete Subgraph*, Analele Universitatii Spiru Haret, Seria Matematicã-Informaticã, anul VII, nr. 2, Editura Fundatiei România de Mâine, Bucuresti 2012.
4. Berge, C., *Graph Theory and Application*, romanian edition, "Editura Tehnica", Bucharest, 1971.
5. Popescu, D.R., *Combinatorica si teoria grafurilor*, "Editura Societatii de Stiinte matematice din Romania", Bucharest, 2005.
6. Tomescu, I., *Combinatorica si teoria grafurilor*, "Editura Universitatii Bucuresti", Bucharest, 1990.