A polynomial-Time Algorithm for the Maximum Clique Problem

Detailed overview

Zohreh O. Akbari

April 6, 2021

I. Introduction

An important consequence of the Cook-Levin theorem claims, that if any NP-complete problem can be solved in polynomial time, then every problem in NP has a polynomial-time solution. Thus besides many important applications in different domains, a polynomial-time solution to the maximum clique problem, as an NP-complete problem causes every problem in NP to have a polynomial solution, which leads to the equality of P and NP complexity classes.

This paper presents a polynomial-time algorithm for the *maximum clique problem*, which detects the maximum clique of a given graph through a recursive approach.

II. The Maximum Clique Problem

Definition 1 (Graph)**.**
Let V be a set of vertices and $E \subseteq \{\{i,j\} \mid i,j \in V, i \neq j\}$ a set of edges. The pair $G = (V,E)$ is called a graph.

Definition 2 (Complete graph)**.**
A graph $G = (V,E)$ is called complete graph, if every pair of distinct vertices is connected by a unique edge. That means $\forall\ i,j \in V,\ i \neq j \Longrightarrow (i,j) \in E$. A graph is complete, if and only if every vertex has degree $|V|-1$. All complete graphs are their own maximal cliques.

The author defines the complete graph by φ.

Definition 3 (Clique)**.**
Let $C \subseteq V$. C is called a clique, if and only if $G = (C,E')$ is complete.

Definition 4 (Maximum clique)**.**
The maximum clique of G is called $\omega(G)$. That means

$$\omega(G) = \max\{|S| : S \text{ is a clique in } G\}$$

Definition 5 (Largest subgraph of G in which α exists)**.**
Let α be the vertex of lowest degree:

$$\alpha = \{j \in V : deg(j) \text{ is minimal}\}$$

The author calls the subgraph, which contains α and all connected vertices to α, the

largest subgraph of G in which α exists.

(α and all his neighbours including edges)

III. A Polynomial-Time Algorithm for the Maximum Clique Problem

Pseudocode:

```
MaxClique(G) {
    if (G is a complete graph)
        for each vertex of G: v
            if (|V| - 1 > max C[v])
                max C[v] := |V|;
                make max CP[v] point to a linked list containing V;
    else
    find the vertex of lowest degree: α
    find the largest subgraph of G in which α exists: G'(V',E')
    MaxClique(G');
    if (V - α ≠ φ)
        MaxClique(G - α);
}
```

Auhtors example:

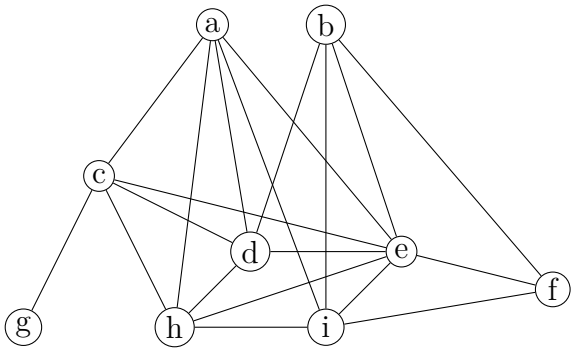


Figure 1: Graph G

We start with this arbitrary graph G.
G is not complete. Therefor we choose the vertex of minimal degree, which is g. That means $\alpha := g$.
Now we determine the largest subgraph G' of G, which contains α.

$$G' = (\{c, g\}, \{\{c, g\}\})$$

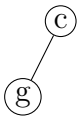


Figure 2: Largest subgraph G' which contains g

G' is complete. Therefor we updated the list and linked list, which was empty. The vertices c and g have value 2 and point to the linked list, which contains the clique we found.

	a	b	c	d	e	f	g	h	i
maxC			2				2		
maxC									

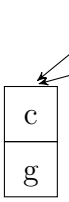


Figure 3: Updated list

Now we remove α from the graph G and we can see that $G - g$ is not complete.

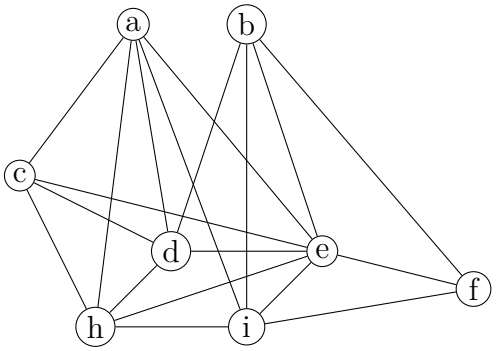


Figure 4: $G - g$

Therefor we obtain the vertex of minimal degree, which is f. That means $\alpha := f$. The largest subgraph, which contains f looks like follows:

$G' = (\{b, e, f, i\}, \{\{b, e\}, \{b, f\}, \{b, i\}, \{e, f\}, \{e, i\}, \{f, i\}\})$

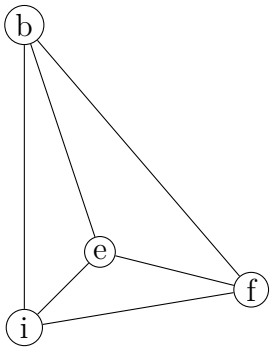


Figure 5: Largest subgraph G' which contains f

G' is complete and we can update the list.

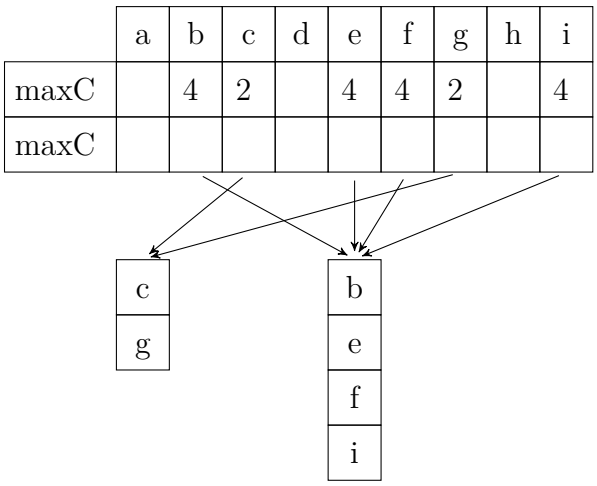


Figure 6: Updated list

Now we remove α from the graph G and we can see that $G - f$ is not complete.

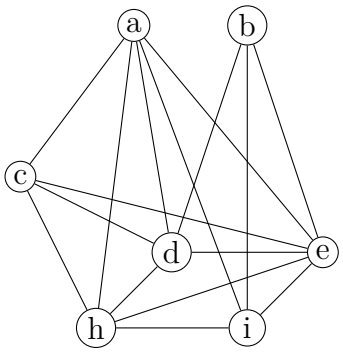


Figure 7: $G - f$

Therefor we obtain the vertex of minimal degree, which is b. That means $\alpha := b$. The largest subgraph, which contains b looks like follows:
$G' = (\{b, d, e, i\}, \{\{b, e\}, \{b, d\}, \{b, i\}, \{d, e\}, \{e, i\}\})$

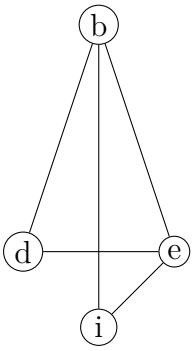


Figure 8: Largest subgraph G' which contains b

Now we can see that G' is not complete. Therefor we repeat the algorithm on this graph

There are two vertices with the same minimal degree, which is two. We can choose d or i. For such case there is no explanation in the paper. We assume that one can choose the vertex at random. In this case we choose $\alpha := d$. Later on there will be the same example but with choice for i.

$G' = (\{b, d, e\}, \{\{b, e\}, \{b, d\}, \{d, e\}\})$

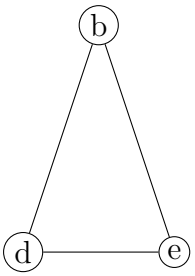


Figure 9: Largest subgraph G' of graph in Figure 8 which contains d

G' is complete and we can update the list. b and e have value four. They will not be updated, because the condition $|V| - 1 > \max C[v]$ is not satisfied. In this case $|V| - 1 = 2$ and $C[b] = 4$ also $C[e] = 4$. That is the reason, why only d is pointing to the new clique.

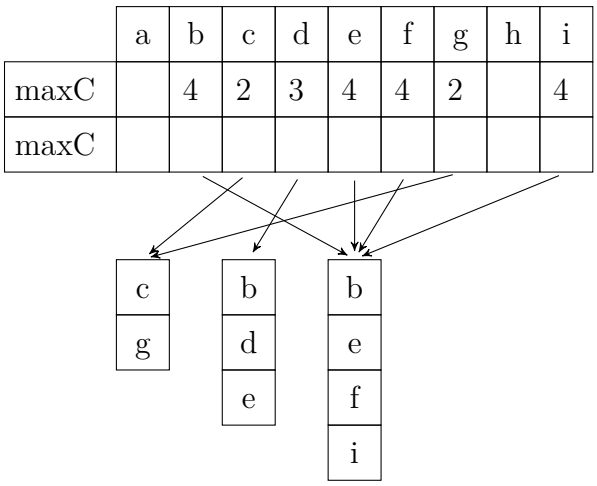


Figure 10: Updated list

Now we have a look on the graph in Figure 8. We remove the vertex d and see that the new graph is complete. $(G' - d) = (\{b, e, i\}, \{\{b, e\}, \{b, i\}, \{e, i\}\})$.

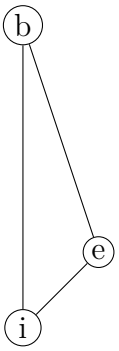


Figure 11: Graph in Figure 8 with removed d

In this case the conditions for all vertices in the clique in Figure 11 are not satisfied. Therefor there is no vertex which has to be updatetd. The list and linked list stay the same as before.

Now we remove α from the graph G and we can see that $G - b$ is not complete.

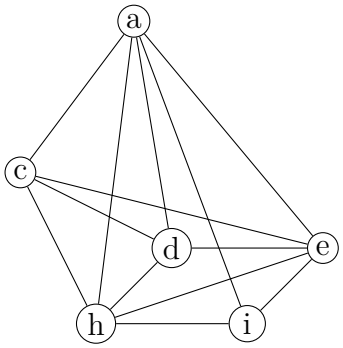


Figure 12: $G - b$

We obtain the vertex of minimal degree, which is i. That means $\alpha := i$. The largest subgraph, which contains i looks like follows:
$G' = (\{a,e,i,h\},\{\{a,e\},\{a,i\},\{a,h\},\{e,i\},\{e,h\},\{i,h\}\})$

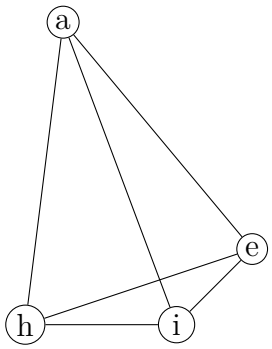


Figure 13: Largest subgraph G' which contains i

We can see that the graph in figure 13 is complete. Therefor it is a clique and the list will be updtated as follows:

	a	b	c	d	e	f	g	h	i
maxC	4	4	2	3	4	4	2	4	4
maxC									

c	b	b	a
g	d	e	e
	e	f	i
		i	h

$G-i$ is a complete graph.

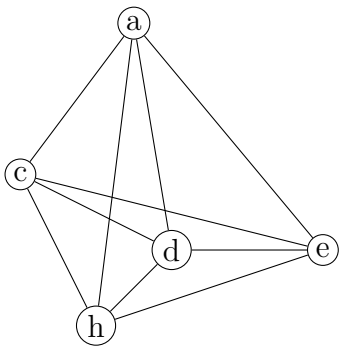


Figure 14: $G-i$

The next and last step is to update the list containing the cliques.
Now we have a problem at codeline 4. Let us look more closely on the if-condition:

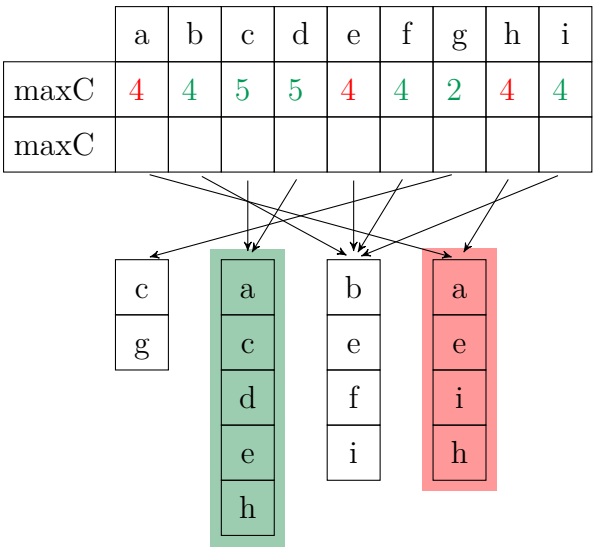
```
if (|V| - 1 > max C[v])
```

Which vertices of our clique in figure 14 satisfy the condition?

$|V|-1=5-1=4$

- $C[a]=4 \not> 4$
- $C[c]=4 > 2$
- $C[d]=4 > 3$
- $C[e]=4 \not> 4$
- $C[h]=4 \not> 4$

The vertices a,e and h will not be updated. This is a mistake, because the vertices in the list must be part of the the maximum clique found for each vertex. In this case the maximum clique found is correct but the list and linked list are not.



Maximum clique contains 5 vertices: $\{a, c, d, e, h\}$

Figure 15: Incorrect list and linked list

The red values represent the error. The green values are correct. The red colored clique should not be stated anymore. The vertices a and h still point to this clique because the condition is not satisfied.

Our assumption is that the codeline 4 must be adjusted as follows:

`if (|V| - 1 > max C[v])` to `if (|V| > max C[v])`

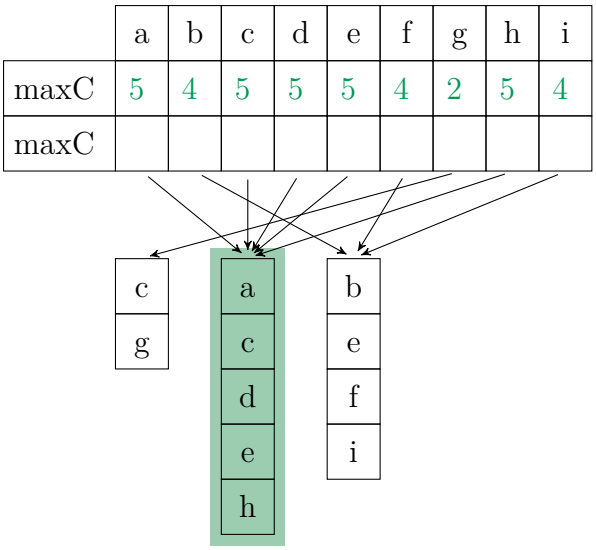
Let us repeat the last step of the example with the adjusted code. In this case we get the authors solution and the correct list:

```
if (|V| > max C[v])
```

$|V| = 5$

- $C[a] = 5 > 4$
- $C[c] = 5 > 2$
- $C[d] = 5 > 3$
- $C[e] = 5 > 4$
- $C[h] = 5 > 4$

All vertices satisfy the updated if-condition. The list looks like as follows:



Maximum clique contains 5 vertices: $\{a, c, d, e, h\}$

Figure 16: Correct list and linked list

Now we get the authors solution and the algorithm stops. The maximum clique found (in both cases) is $\{a, c, d, e, h\}$.