CSC721 Algorithms Fall 2017

Theorem: The k-clique problem is $\mathcal{NP}$-complete.

Definition: Let $G = (V, E)$ be an undirected graph.

- A k-clique of G is a subgraph with k vertices that are completely connected. I.e., for any two vertices v and w in the k-clique, the edge (v, w) is in E.
- The k-clique problem can be stated as follows: Given an undirected graph $G = (V, E)$ and a positive integer k, where $3 \leq k \leq |V|$, does G have a k-clique ?

Theorem: The k-clique problem is $\mathcal{NP}$-complete.

Proof: The idea of the proof is to construct a polynomial time mapping reduction from CNF.[1]

Let

$$F = F_1 \wedge F_2 \wedge ... \wedge F_q$$

be a boolean expression in conjunctive normal form with q factors. Recall by definition of CNF that each factor in F is of the form:

$$F_i = x_{i,1} \vee x_{i,2} \vee ... \vee x_{i,k_i}$$

where each $x_{i,j}$ is a literal[2]

We construct a graph G such that G has a q-clique if and only if F is satisfiable.

The vertices of the graph consist of a pair of integers (i, j) where the first component, i corresponds to a factor in F, and the second component j corresponds to a literal in F_i. Formally, we can define our vertex set as:

$$V = \{\ (i, j) \mid 1 \leq i \leq q \quad \text{and} \quad 1 \leq j \leq k_i\ \}$$

The key idea to keep in mind is that a vertex represents a literal in some factor of the boolean expression F.

The edges of G are defined as follows:

An edge exists between two vertices, (i, j) and (k, ℓ) if and only if:

1. $i \neq k$ and
2. it is possible to assign values to the variables in the literals $x_{i,j}$ and $x_{k,\ell}$ in such a way that both literals are true.

Condition 1 above asserts that the edge connects vertices that correspond to different factors in F.

Condition 2 above asserts that an edge connects vertices that correspond to literals that either:

1. contain different variables, **or**
2. contain the same variable and neither variable is complemented **or**
3. contain the same variable and both variables are complemented

[1] Here, CNF denotes the set of satisfiable boolean expressions in conjunctive normal form.

[2] Recall that a literal is a boolean variable x, or its complement $\neg x$.

Now that we have defined G, we make the following claim.

Claim: G has a q-clique if and only if F is satisfiable.

"⇒" Suppose G has a q-clique. Let

$$\{\ (i_1, j_1), (i_2, j_2), \ldots, (i_q, j_q)\ \}$$

denote the set of vertices in the q-clique. Notice that the first component of every vertex is unique. I.e., $i_1, i_2, \ldots, i_q$ are all distinct.[3] If we sort the vertices by first component, we can re-write them in an ordered list:

$$(1, \hat{j}_1), (2, \hat{j}_2), \ldots, (q, \hat{j}_q)$$

All of the integers 1 to q are present in the first components in list above, therefore there is a one-to-one correspondence between vertices in the clique, and the factors comprising F. I.e., vertices $(1, \hat{j}_1), (2, \hat{j}_2), \ldots, (q, \hat{j}_q)$ directly correspond to the literals

$$x_{1,\hat{j}_1}\ ,\ x_{2,\hat{j}_2}\ ,\ \ldots,\ x_{q,\hat{j}_q} \tag{1}$$

Recall that by the definition of G, edges exist between two vertices if and only if there is an assignment of values to the variables such that both of the corresponding literals are true. Every pair of vertices is directly connected in the clique, thus there is an assignment of values to the variables such that each of the literals in the list above is simultaneously true. One literal in each factor of F is true and therefore, F is satisfiable.

"⇐" Suppose F is satisfiable. Let

$$x_{1,\hat{k}_1}\ ,\ x_{2,\hat{k}_2}\ ,\ \ldots,\ x_{q,\hat{k}_q} \tag{2}$$

denote a list of literals which can be made true by some assignment of values to the variables. Let

$$(1, \hat{k}_1)\ ,\ (2, \hat{k}_2)\ ,\ \ldots,\ (q, \hat{k}_q) \tag{3}$$

denote the list of vertices corresponding to the literals in list (2). By the definition of G, there is an edge between every pair of vertices in the list (3). Therefore, the set of vertices in the list (3) forms a q-clique.

Finally, observe that the construction described above can be implemented by a polynomial time deterministic Turing machine. Therefore there exists a polynomial time computable function that maps CNF-satisfiability to the k-clique problem. We have already shown that CNF-satisfiability is $\mathcal{NP}$-complete, therefore the k-clique problem is also $\mathcal{NP}$-complete.

[3] Why is this true ? Because if there were two vertices with the same first component, there could not be an edge between them (by definition of the edge set in G). But if there is no edge between the two vertices, they can not be part of a clique.