On the k-clique Problems: A New Approach

Dhananjay P. Mehendale
Sir Parashurambhau College, Tilak Road, Pune-411030,
India.

Abstract

In this paper we discuss new approach to deal with *k*-clique problems or their equivalents, namely, *k*-independent set problems.

1. **Introduction:** A graph is **complete** when every vertex of it is connected by an edge to every other vertex and a graph is **totally disconnected** when there are no edges joining any of its vertices. A **clique** in a graph is its subgraph which is complete. A **maximal clique** is one in which no vertices can be added. In other words, a maximal clique is one which is not included in a larger clique, i.e. which is not a proper subgraph of a larger clique. It is not difficult to see that finding maximal clique is that way quite straightforward. A **maximum clique** is clique with largest possible number of vertices. **Clique number** associated with a graph, *G*, $\omega(G)$, is equal to number of vertices in a maximum clique of *G*. Finding maximum clique in *G* is not easy, and rather a hard problem. A brute force algorithm to check whether *G* contains a *k*-clique is in the worst case may require to test all subgraphs of *G* containing *k* vertices and see whether there is some subgraph with *k* vertices among these subgraphs which is clique. Solving the **decision clique problem** of testing whether a graph contains a clique larger than given size is NP complete. Input: Undirected graph *G* and a number *k*. Output: Boolean value true if contains a clique of size *k*, and false otherwise. The simplest nontrivial case of clique finding is finding triangle in a graph, or equivalently, to determine whether the graph is triangle free. The problem of finding the maximum clique in a graph is NP hard. Input: Undirected graph *G*. Output: maximum sized clique in this graph *G*. The **complement** of a graph is the one obtained by replacing edges by non-edges and vice versa. The **independent set** in a graph is a set of vertices which together forms totally disconnected subgraph of *G*. A **maximal independent set** is one in which no new vertex can be added keeping it independent. A **maximum independent set** is independent set with largest possible number of vertices. The **clique problems** and the **independent set problems** are equivalent as they are complementary.

The equivalence of these problems is straightforward: for a clique in G is same as independent set in the complement of G, say G^c. Thus, the decision independent set problem is NP complete. Input: Undirected graph G and a number k. Output: Boolean value true if contains an independent set of size k, and false otherwise. The maximum independent set problem is NP hard. Input: Undirected graph G. Output: maximum independent set in the graph.
We will consider **clique problems** in this paper.

2. **Turan and Turan-like Graphs and their Subgraphs:** In the so called problems of extremal graph theory one asks to find out $ex(p, H)$, the count of maximum number of edges a graph on p vertices can contain, without containing the forbidden graph H. The following celebrated theorem of Turan is forerunner of the field of extremal graph theory. Let $\lfloor r \rfloor$ denotes the greatest integer not exceeding the real number r and let $\lceil r \rceil$ denotes the smallest integer not less than the real number r. Turan [1] showed that the maximum number of edges among all graphs containing p vertices and without containing any triangles is $\left\lfloor \frac{p^2}{4} \right\rfloor$. We give below the standard definition of Turan graph [2] and define further Generalized-Turan graph or Turan-like graph.

Definition 2.1: The Turan graph $T_{n,r}$ is the complete r-partite graph with n vertices and has b parts of size $a+1$ and $r-b$ parts of size a, where $a = \left\lfloor \frac{n}{r} \right\rfloor$ and $b = n - ra$.

Turan [1] proved that $T_{n,r}$ is the unique largest simple n vertex graph with no clique on $(r+1)$ vertices, i.e. $T_{n,r}$ doesn't contain any $(r + 1)$-clique.

Theorem 2.1(Turan): Among the n-vertex simple graphs with no $(r + 1)$-clique, $T_{n,r}$ has maximum number of edges. □

Definition 2.2: The Generalized-Turan graph or Turan-like graph, $L_{n,r}$ is the complete r-partite graph with n vertices and has r parts of all possible various sizes such that the sum of cardinalities of all these parts is n.

Thus, in order to form a Turan-like graph we take a set of *n* vertices. We form some *r*-partition of number *n*, as follows:

$$n = p_1 + p_2 + p_3 + \cdots + p_r$$

We now split the initially taken *n* vertices into *r* number of subsets of vertices such that the cardinalities of these subsets is $p_1, p_2, \cdots, p_r$ respectively, and we then form the *r*-partite complete graph. It is also clear to see that the Turan graph $T_{n,r}$ is a special type of Turan-like graph $L_{n,r}$ only.

If now we drop the condition of completeness possessed by Turan graph $T_{n,r}$ or Turan-like graph $L_{n,r}$ then what we will get will be different **proper subgraphs** of these graphs.

An important point to be noted is as follows: No graph having representation as complete or incomplete *r*-partite graph, i.e. having representation as $T_{n,r}$ or $L_{n,r}$ or some proper subgraph of these graphs has (*r*+1)-clique as subgraph because each partite set can contribute at most one vertex to a clique.

Let *G* be a (p, q) graph, i.e. a graph on p points (vertices) and q lines (edges) with the following vertex set $V(G)$ and edge set $E(G)$ respectively:

$$V(G) = \{v_1, v_2, \cdots, v_p\} \text{ and}$$
$$E(G) = \{e_1, e_2, \cdots, e_q\}$$

Definition 2.3: The vertex adjacency bitableau associated with every labeled copy of graph *G*, *VAB*(*G*), is the following bitableau:

$$VAB(G) = \left(\begin{array}{c|cccc} 1 & \alpha_1^1 & \alpha_2^1 & \cdots & \alpha_{l_1}^1 \\ 2 & \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_{l_2}^2 \\ \vdots & & & & \\ j & \alpha_1^j & \alpha_2^j & \cdots & \alpha_{l_j}^j \\ \vdots & & & & \\ p & \alpha_1^p & \alpha_2^p & \cdots & \alpha_{l_p}^p \end{array}\right)$$

where left tableau represents the suffixes of the vertex labels and stand for the vertices while the right tableau represents the rows of the suffixes of the vertex labels and represent the vertices that are adjacent to vertex whose suffix is written in the same row in the left tableau, i.e. the appearance of entry α_k^j in the j^{th} row of the right tableau implies that vertex v_j is adjacent to vertex $v_{\alpha_k^j}$.

Definition 2.4: The **increasing vertex adjacency bitableau** associated with every labeled copy of graph G, $IVAB(G)$, is the following bitableau:

$$IVAB(G) = \left(\begin{array}{c|ccccc} 1 & 2 & 3* & \cdots & p \\ 2 & 3 & 4* & \cdots & p* \\ \vdots & & & & \\ j & (j+1)* & (j+2) & \cdots & p \\ \vdots & & & & \\ p & - & & \cdots & \end{array} \right)$$

where left tableau represents the suffixes of the vertex labels and stand for the vertices while the right tableau represents the rows of the suffixes of the vertex labels and represent the vertices that are adjacent (nonadjacent) to the vertex whose suffix is written in the same row in the left tableau, i.e. the appearance of **entry k ($k*$) in the j^{th} row** of the right tableau implies that $j < k$ and vertex v_j is **adjacent (nonadjacent)** to vertex v_k.

Thus, for example, from the above given $IVAB(G)$ vertex v_1 is adjacent to vertex v_p, but vertex v_j is not adjacent to vertex $v_{(j+1)}$, etc. It is clear to see that for a complete graph, K_p, on p points

$$IVAB(K_p) = \left(\begin{array}{c|ccccc} 1 & 2 & 3 & \cdots & p \\ 2 & 3 & 4 & \cdots & p \\ \vdots & & & & \\ j & (j+1) & (j+2) & \cdots & p \\ \vdots & & & & \\ p & - & & \cdots & \end{array} \right)$$

Similarly, for an independent set, I_p, on p points

$$IVAB(I_p)=\left(\begin{array}{c|cccc} 1 & 2* & 3* & \cdots & p* \\ 2 & 3* & 4* & \cdots & p* \\ \vdots & & & & \\ j & (j+1)* & (j+2)* & \cdots & p* \\ \vdots & & & & \\ p & - & & \cdots & \end{array}\right)$$

As an illustration consider the $IVAB(G)$ constructed for a graph on $p = 7$ points:

$$IVAB(G)=\left(\begin{array}{c|cccccc} 1 & 2 & 3 & 4* & 5 & 6 & 7* \\ 2 & 3 & 4 & 5* & 6 & 7 & \\ 3 & 4 & 5 & 6* & 7 & & \\ 4 & 5 & 6 & 7* & & & \\ 5 & 6 & 7 & & & & \\ 6 & 7 & & & & & \\ 7 & - & & & & & \end{array}\right)$$

Clearly, this graph is $T_{7,3}$, a complete 3-partite graph with independent sets {1, 4, 7}, {2, 5}, {3, 6} and so it can't contain a clique on $n = 4$ points.

We are going to state now an obvious but important result:

Theorem 2.2: Every simple graph on *n* vertices can be shown to be equivalent to (isomorphic to) either some $T_{n,r}$ or $L_{n,r}$ or some proper subgraph of these graphs, for some *r*. This representation for the graph under consideration is not unique and many different representations in terms of $T_{n,r}$ or $L_{n,r}$ or some proper subgraph of these graphs are possible for the graph under consideration.

Proof: This result follows if we see that we can break the vertex set of given graph into subsets which are independent sets. Thus, we need to see that the vertex set of any graph can be expressed as disjoint union of independent sets. Let {1, 2, **….**, *n*} be the labels associated with the vertices of the graph under consideration. To form first independent set we take vertex with label 1 as first element of first independent set.

We then find and add vertex with smallest label, say j_{11}, which is nonadjacent to vertex with label 1 to this set. We then find and add vertex with smallest label, say j_{12} which is nonadjacent to vertex with label 1 and j_{11} to this set. We then find and add vertex with smallest label, say j_{13} which is nonadjacent to vertex with label 1, j_{11} and j_{12} to this set. In this way we go on adding vertices till possible and form first independent set $\{1, j_{11}, j_{12}, j_{13}, \cdots\}$ such that all the vertices in this set are mutually nonadjacent. We then choose vertex with smallest label which is not present in just formed first independent set and taking this vertex as first vertex in the second independent set to be formed we proceed on similar lines and form second independent set. We continue this procedure of forming independent sets till every vertex among the vertices with labels {1, 2, **….**, *n*} belonging to graph under consideration will belong to some independent set. The result is now clear.

□

It is easy to see that an efficient algorithm to express given graph *G* as an *r*-partite graph with **minimum *r*** provides efficient solution to the *k*-clique problem. In this representation of *G* as an *r*-partite graph, with *r* minimum, this graph may or may not be complete *r*-partite graph. Thus, in brief the given graph *G* is expressed in terms of either some $T_{n,r}$ or $L_{n,r}$ or some proper subgraph of these graphs with minimum value for *r*. Such graph will certainly not contain a clique of size (*r*+1), i.e. $K_{(r+1)}$, and may or may not contain clique of size *r* , i.e. K_r, because the *r*-partite graph thus formed is not necessarily complete and may be proper subgraph of complete *r*-partite graph. When this graph will be the complete *r*-partite graph then it will certainly contain a clique of size *r*, i.e. a K_r. We now proceed to give one algorithm to manage the task of expressing given graph *G* as an *r*-partite graph with ***r* minimum.**

Algorithm 2.1:

1) Given the graph *G* with vertices labeled {1, 2, 3, … , *n*}. Form all possible maximal independent sets which contain vertex with label 1.

2) For each maximal independent set thus formed containing vertex labeled 1, say set $A_1, A_2, \cdots$ etc. Find the smallest vertex labels say $j_{11}, j_{21}, j_{31}, \cdots$ such that vertex with label $j_{i1} \notin A_i$ for all *i*.

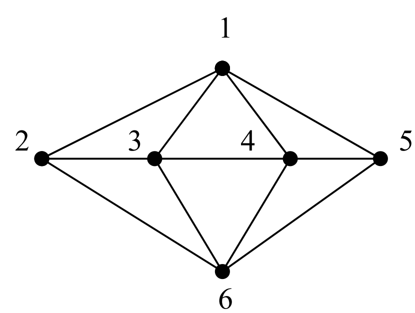
3) Form all possible maximal independent sets containing vertex with label j_{11}, say $A_{11}, A_{12}, A_{13}, \cdots$ also form all possible maximal independent sets containing vertex with label j_{21}, say $A_{21}, A_{22}, A_{23}, \cdots$ and so on. Continue this way till each of the vertex is placed in some independent set. In this way we have formed different collections of independent sets such that each such collection of independent sets together contains all the vertices.

4) Among these different collections of independent sets, each representing the given graph as an *r*-partite graph for some *r*, find those which are having smallest value for *r*.

5) Collect all these representations with smallest value for *r* which represent the expressions for given graph *G* as an *r*-partite graph with ***r* minimum.**

□

Example: Consider following simple graph



It is easy to check that by following the steps of the above algorithm we can form the following two collections of independent set as representation for the graph in the above figure as 3-partite graph with partite sets: {1, 6}, {2, 4}, {3, 5}. Or it can be represented as 4-partite graph with partite sets: {1, 6}, {2, 5}, {3}, {4}. It can be checked that the representation as 3-partite graph is minimal, i.e. the minimum value for *r* is 3.

One should now note an **important point** at this juncture that if and when we get the representation for the graph under consideration as some *r*-partite graph, may be complete *r*-partite one or not, then this graph cannot contain as subgraph some (*r*+1)-clique, and even if by adding the required missing edges we convert this *r*-partite graph into complete r-partite graph still such new graph will not contain as subgraph the $(r+1)$-clique.

3. **Utilizing IVAB(*G*) for Clique Finding:** We now discuss a **heuristic** algorithm in order to quickly locate a clique of certain size in a graph as its subgraph. We make use of IVAB(G) of given graph G. The k-th row ($k < n$) of IVAB(G) for graph G containing n vertices is

$$k \mid (k+1)\ \ (k+2)^*\ \ (k+3)\ \ (k+4)\ \ (k+5)^* \ldots\ldots\ldots n$$

Thus, it contains all numbers from $(k+1)$ to n, in all $(n-k)$ numbers, and those numbers which represent labels of vertices nonadjacent to k are marked with a * after these numbers. Now suppose we wish to check whether some clique of size m is present in given graph G. We follow the following procedure:

Algorithm for Clique Testing:

1) Form IVAB(G) for given graph G.

2) Associate number $\alpha(k)$ with each row k, where $\alpha(k) = \dfrac{\beta(k)}{(n-k)}$, and $\beta(k)$ stands for the count of entries in k-th row which are not marked by a star after them.

3) Collect those m rows which contains maximum number of entries not marked with star, i.e. collect those rows for which $\alpha(k)$ is large say $\{i_1, i_2, i_3, \cdots, i_m\}$. You may arrange the numbers $\alpha(k)$ associated with rows in non-increasing order and take first m rows with large $\alpha(k)$.

4) Check whether in row i_1 numbers $\{i_2, i_3, \cdots, i_m\}$ are not marked with * after them, in row i_2 numbers $\{i_3, \cdots, i_m\}$ are not marked with * after them, ……., in row $i_{(m-1)}$ if number $\{i_m\}$ is not marked with * after it.

□

Remark 3.1: When the check in step 4) above has positive answer then clique formed by vertices with labels $\{i_1, i_2, i_3, \cdots, i_m\}$ exists as subgraph of G.

Remark 3.2: When one wish to check whether given graph contains a triangle, or whether it is triangle free then in the bitableau IVAB(G) for the given graph G one needs to check whether there exists or not a sub-bitableau of the form

$$\begin{pmatrix} i_1 \mid & i_2 & i_3 \\ i_2 \mid & i_3 & \end{pmatrix}$$

Remark 3.3: When one wish to check whether given graph contains an independent set of size 3, or whether it is free of any 3-independent set then in the bitableau IVAB(G) for the given graph G one needs to check whether there exists or not a sub-bitableau of the form

$$\begin{pmatrix} i_1 \mid & i_2 * & i_3 * \\ i_2 \mid & i_3 * & \end{pmatrix}$$

References

1. Turan P., Eine Extremalaufgaba aus der Graphentheorie. Mat. Fiz Lapook, 48, 436-452, 1941.
2. West Douglas B., Introduction to Graph Theory, Prentice-Hall of India, Private Limited, New Delhi 110-001, 1999.