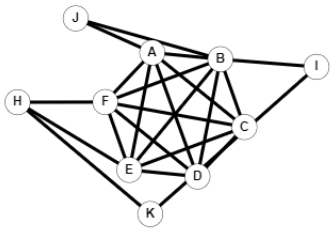
Clique

Teja Kalvakolanu

April 2020

1 Problem Definition

Given a graph $G = (V, E)$, a clique in the graph G is a complete sub graph with vertices K such that $K \subseteq V$, every two vertices in K are two end points of an edge in E.
For example the graph below has vertices $V = \{A, B, C, D, E, F, G, H, I, J\}$



The cliques that could be formed are
$\{A, B\}$,$\{A, B, C\}$,$\{A, B, C, D\}$,$\{A, B, C, D, E\}$,$\{A, B, C, D, E, F\}$
But we would be most interested in finding the clique $\{A, B, C, D, E, F\}$ since it is maximal.

Maximal Clique
Input: A graph $G = (V, E)$
Goal: Return Maximal Clique

The above problem is the optimization version of the clique problem which is finding the maximal clique. This can be converted to the decision problem with a little modification

Decision Problem For Clique
Input: A graph $G = (V, E)$ and a variable K.
Goal: Return Yes if there exists a clique of size K or else NO.

We will now try to show that the decision version of Clique is actually a Decision Problem by which we can prove that the $Clique \in NP$

2 Clique ∈ NP

We will try to prove by designing an Algorithm that takes an instance of Clique and a solution and checks whether the solution is right or wrong.

CHECKCLIQUE(I=$(G(V, E), k)$, S)

Input: An Instance of Clique: A graph $G = (V, E)$ with a positive integer k with a proposed solution $S \subseteq V$.
Output: True if S is a solution for I and False if it is not.

```
1: if |S| ≥ k then
2:     for all u ∈ S and v ∈ S − {u} do
3:         if (u, v) ∉ E then
4:             return False
5:     return True
6: return False
```

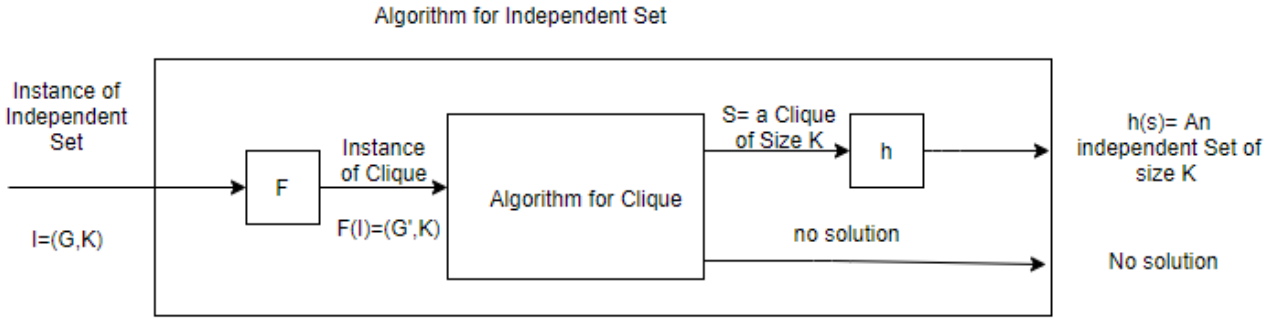
This checking algorithm will check the solution S with a run time of $O(|S|^2)$(Certainly polynomial).
The run time is because we need to check all possible of combinations of vertices in S. Since Clique Checking algorithm is Polynomial run time Clique $\in NP$.

3 Clique ∈ NP-complete

To prove that Clique is NP-complete we provide two reductions.

3.1 Independent Set $\leq_p$ Clique

First we reduce Independent Set to Clique and prove that there exists a polynomial time algorithm



Given an Instance of Independent Set I=(G,K) we need to transform it in to the instance of Clique ($\overline{G}$,K) using function F. The solution of the Clique S should be converted in to the solution of independent set Using the function h

- F($I = (G, K)$)=($\overline{G}, K$) where $\overline{G}$ is the complement of G. The F function will convert the graph into its complement.
- h(S)=S , The solution obtained by the clique for the complement graph $overline G$ is the Independent Set for graph G . if F(I) has no solution then I also has no solution.

The reduction will be correct if the following theorem is True

Theorem 1 *A set of Vertices S is Independent Set of graph G if and only if S is a Clique of $\overline{G}$ and vice versa.*

Proof:

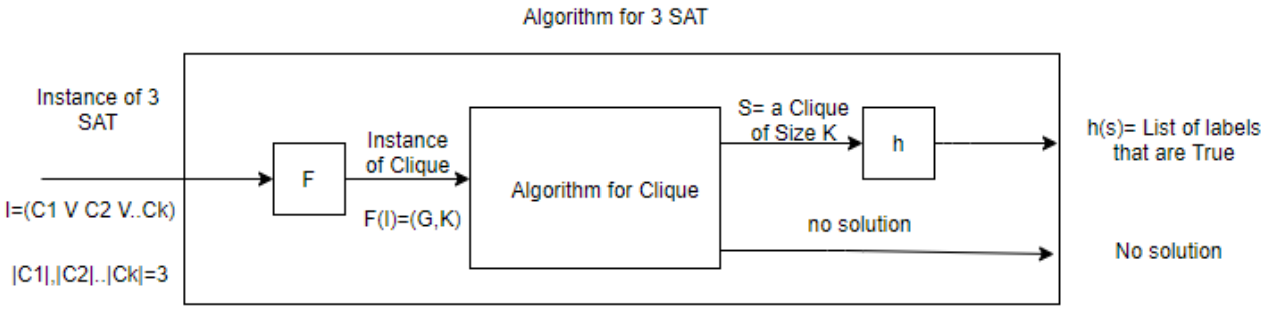
- Suppose S is an independent set of G and by proof of contradiction S is not a clique of $\overline{G}$
- Then $\exists e = (x, y)$ in S such that there is no edge $e \in E$, Then $e \in E$ which is a contradiction to S is an independent set of G.
- Suppose S is a clique of $\overline{G}$ and for contradiction S is not an independent set of G.

- Then $\exists e = (x, y)$ in S such that $e \in E$. The $e \notin \overline{E}$ which is a contradiction to S is a Clique of $\overline{G}$.

■

3.2 3 SAT $\leq_p$ Clique

we reduce 3 SAT to clique to prove that Clique is NP complete Given an instance of 3 SAT I we transform it in to instance of Clique (G, K) using Function F. The solution of the clique S is converted in to the Solution of the independent set Using the function h.



- $F(I) = (G, K)$. The function F will convert the 3 SAT in to (G, K) by following conditions
  - Let m be the number of clauses of I.Set k=m and make a graph G with m clusters of up to 3 nodes each.
  - Each cluster corresponds to a clause of I. Each node in a cluster is labeled with a literal from the clause.
  - do not connect any nodes in the same cluster, add edges between all pairs of nodes in different clusters, except pairs of the form x, $\overline{x}$
- h(S) takes all the vertices of the S and sets their value to True. This solves the 3 SAT. if F(I) has no solution then I also has no solution.

The Reduction will be correct if we prove the following Theorem.

Theorem 2 *if I is an instance of 3 SAT with K clauses then there exists a clique of size atleast K in graph G*

Proof:

- Given a SAT assignment for every clause there exists at least one literal that is set true.
- Let V_c be the vertex in G corresponding to the first literal of Clause C that was set True.
- we have to prove that $V_{C_1}, V_{C_2} \dots V_{C_K}$ is a K-clique to prove the theorem
- if (V_{c_1},V_{c_2}) $\notin E$ then they both must be of form X, $\overline{x}$. But both of them cannot be true. There fore (V_{c_1},V_{c_2}) $\in E$ for all $V_{c_1}, V_{c_2} \in S$. So, S has an K-clique.

■

Theorem 3 *Given a graph G with a clique of size K which is solution S then we construct a SAT assignment which is satisfiable.*

Proof:

- The solution for 3 SAT has exactly 1 node from each cluster.
- For each variable x of 3 SAT assigned True there exists a vertex in S.
- For every clause C, one vertex from cluster C is in S. Therefore at least one variable in a clause is True.
- Therefore the whole 3 SAT is satisfiable.

■

4 Brute Force Algorithm

BRUTEFORCECLIQUE(I=($G(V, E), k$))

Input: An Instance of Clique: A graph $G = (V, E)$ with a positive integer k.
Output: Returns a Set of k vertices $S \subseteq V$ that forms a clique or no solution.

```
1: for all subsets R ⊆ V of size k do
2:     if R is a clique then
3:         return R
4: return "No Such Clique Exists"
```

To select R of size k for vertices V it costs $\binom{n}{k}$ if $|V| = n$. It Costs O(k^2) to check if it is a Clique. The total run time for Brute Force Clique Algorithm is O($k^2\binom{n}{k}$).

5 Approximation Algorithms

There is no approximation algorithms for Clique. Hence we derive a non trivial approximation of clique with approximation ratio of $O(n/logn)$. We use the optimal version of the Clique problem which is the Maximal Clique.

Maximal Clique
Input: A graph $G = (V, E)$
Goal: Return Maximal Clique

Unfortunately, this problem does not lend itself to approximation.To attain an approximation ratio better than $O(n^{0.99})$ is NP-complete. In other words, if the maximum clique is K^* we will not be able to find a clique K such that $n^{.99}|K| \geq |K^*|$. This is a very discouraging result.

But there exists a non trivial approximation where we use the fact that a subset of a clique is also a clique and derive the approximation algorithm.

CLIQUEAPPROXIMATION(I=($G(V, E)$))

Input: An Instance of Perfect Graph: A graph $G = (V, E)$.
Output: A Clique for Graph G.

```
1: Divide the vertices |V| in to n/k groups each of size k where |v| = n.
2: for S in n/k Groups do
3:     for all subsets R ⊆ S do
4:         if R is a clique then
5:             store R in C
6: return max(C)
```

This takes $O(nk2^k)$ run time. But if k is restricted to $logn$ then the whole run time will be polynomial.
Now we need to prove its feasibility and get the approximation ratio.

Theorem 4 *The CLIQUEAPPROXIMATION Returns a Clique.*

Proof: For contradiction lets assume that it does not return Clique. But according to pigeon hole principle if a graph consists of clique of size k^* when the vertices are divided in to sets of length k each set contains at least $k^*(k/n)$ vertices that are part of the clique. Therefore our assumption that it does not return clique is false. ■

Theorem 5 *The CLIQUEAPPROXIMATION $O(n/logn)$ Approximation.*

Proof:

- This Approximation Can be proved by considering a fact that the subset of clique is also a Clique.
- Let K^* be the maximal clique in the graph. By the pigeonhole principle, at least one of the n/k groups must contain at least $(k/n)|K^*|$ of the vertices of K^*
- Thus, letting K be these vertices, our approximation algorithm finds a clique satisfying $|K| \geq (k/n)|K^*|$.
- This is an approximation ratio of O(n/ log n).

■

6 Applications/ Other interesting Things

- Maximum Clique is encountered in social network communication analysis, computer network analysis, computer vision and many others.
- Clique is proposed as an alternative solution to proof of work scheme for mining the bit coin . By using the clique scheme the miner himself cannot find a largest clique with out knowing the clique size.
- Clique detection has traditionally been a very important tool in chemo-informatics, where one problem is to identify common substructures. The molecules are viewed as (low tree width) graphs and clique detection is used to find common structures (the "cliques" are defined across graphs, not within)
- Another use of clique-detection methods is in cluster analysis of gene expression data. Modern DNA microarray technology can generate simultaneous data on the expression levels of thousands of genes in a cell line or tissue, biologists hope to extract patterns of coordinated changes in expression, as these may reveal new insights into gene networks involved in normal biology or disease.. It is assumed the true clusters can be represented by a set of disjunct cliques, whose union is the clique graph H, and that the input dataset deviates from the true cluster structure due to random errors that result from the noisy, complex experimental process. The input data is first converted to a similarity graph G, where vertices represent genes, and an edge exists between any pair of genes whose expression pattern satisfies a criterion of similarity. Edges and non-edges in G are then flipped with probability until a clique graph is obtained, with the goal of removing the "corruption" from G with as few flips (errors) as possible. When applied to published gene expression datasets, this algorithm (called CAST) recovered clusters similar to those from other methods, and required fewer assumptions by the user. In addition, it was quite fast,running in time $O(n2[logn]c)$.

- Maximal cliques are used in the context of coalition formation among autonomous agents to define an algorithm that allows independent unmanned vehicles to locate other agents and team up with them in an ad-hoc manner to share resources. For example, information that each agent has about its local environment can be shared to create a broader map.
- When the number of agents grows sufficiently large, it is impractical to have a centralized authority appoint a leader or break the agents into cooperating groups. Tosic and Agha provided a method for allowing the agents themselves to choose their affiliation based on distance and utility metrics, provided they can communicate throughout the group-choosing process. Fault tolerance recommends that all agents in a group be able to communicate, so the maximal clique model is a natural one for defining groups. The fluidity of agents moving into or out of range requires that the group-formation algorithm be efficient, as it may be called multiple times during a task set. The authors' algorithm involves several stages in which an agent first identifies the other agents in its neighborhood, then considers the available groups as each of its neighbors chooses an affiliation. As this information comes in, the agent eventually makes its own choice, and will remain affiliated with that group until the task set is complete.
- Trying to avoid detection. Hayes (2006) attempted to reverse-engineer the National Security Agency's telephone surveillance program from descriptions of its goals and knowledge about its targets. Early published accounts of the program made references to "call graphs." While it is impossible to determine precisely the capabilities of the NSA, a maximal clique of small-to-moderate size in a call graph,with little communication outside the clique, might be a plausible flag for a suspicious set of nodes.
- Studies by ATT in the late 1990's studied problems with similar structure, examining the graph that resulted from all of the phone calls across the network in one day. This graph had over 50 million nodes (phone numbers) and 170 million edges (calls between numbers). They used an approximate, probabilistic method to find cliques in the graph. The largest maximal clique they found had 30 vertices, representing 30 people, each of whom talked to all 29 other people in the clique on that day. This experiment revealed a major obstacle in getting useful information out of this analysis, as the analysis discovered more than 14,000 distinct instances of 30-node cliques. Assuming a small number of malicious cells, the large number of false positives would make using cliques as a starting point prohibitively noisy. They could plausibly be used, however, to hone in on a target already under suspicion for other reasons.

References

[1] Bag, Samiran and Ruj, Sushmita and Sakurai, Kouichi, On the Application of Clique Problem for Proof-of-Work in Cryptocurrencies", Information Security and Cryptology","2016".

[2] Satish Rao, Keviln Laker, Bill Kramer, "Combinatorial Algorithms and Data structures", Georgia Tech.