Clique

Prudhvi Varma Penmetcha

December 5, 2011

Outline

- A Clique in an undirected graph $G = (V,E)$ is a subset of the vertex set $C \subseteq V$, such that for every two vertices in C, there exists an edge connecting the two.
- Maximum Clique: A Clique of the largest possible size in a given graph.
- Maximal Clique: A Clique that cannot be extended by including one more adjacent vertex.

- The 'Clique' terminology comes from Luce and perry (1949).
- First Algorithm for solving the Clique problem is that of Harary and Ross (1957).
- Tarjan and Trojanowski (1977), an early work on the worst-case complexity of the Maximum Clique problem
- In the 1990s, a breakthrough series of papers beginning with Feige (1991) and reported at the time in major newspapers, showed that it is not even possible to approximate the problem accurately and efficiently.

- Bron Kerbosch Algorithm
- Backtracking Algorithm

- The Algorithm was Designed and Published in 1973 by the Dutch scientists Joep Kerbosch and Coenradd Bron.
- Bron Kerbosch Algorithm is for Finding the Maximal Cliques in undirected graph.
- It is Known to be one of the most efficient algorithm which uses recursive backtracking to find Cliques is practically proven.
- The Bron Kerbosch Algorithm uses the vertex in graph and its neighbours with few functions to generate some effective results.

Without Pivoting Strategy

Bronkerbosch(R,P,X)
if $\{ P = X = \Phi \}$
Report R as the Maximal Clique
for each vertex v in P
BronKerbosch(R $\cup \{ v \}$,P $\cap$ N $\{ v \}$, X $\cap$ N $\{ v \}$)
$P := P \setminus \{ v \}$
$X := X \cup \{ v \}$

With Pivoting Strategy

Bronkerbosch(R,P,X)

if { $P = X = \Phi$ }

Report R as the Maximal Clique

Choose Pivot Vertex ‘u’ in $P \cup X$

for each vertex v in P

BronKerbosch($R \cup \{ v \}, P \cap N\{ v \}, X \cap N\{ v \}$)

$P := P \setminus \{ v \}$

$X := X \cup \{ v \}$

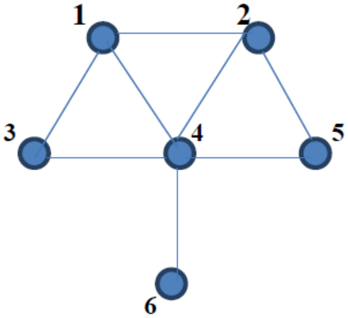

Figure: Example 1

- R=X=Φ P=(1,2,3,4,5,6)
- Choosing the pivot element 'u' as 4.
- 4 in P $\setminus$ N(v) $= (1,2,3,4,5,6) \setminus (1,2,3,5,6) =$ 4 in 4
- Find's the values of P_{new},R_{new},S_{new}
- $P_{new} = \mathrm{P} \cap \mathrm{N(v)}$, $R_{new} = \mathrm{R} \cup (v)$, $X_{new} = \mathrm{X} \cap \mathrm{N(v)}$
- $R_{new} = 4$, $P_{new} = (1,2,3,5,6)$, $X_{new} = \Phi$
- Bronkerbosch(4),(1,2,3,5,6),Φ)

  Bronkerbosch((4,1),(2,3),Φ)

  Bronkerbosch((4,1,2),Φ,Φ)

  Report (4,1,2) as one of the Maximal Clique

- Bronkerbosch((4),(1,2,3,5,6),Φ)
  Bronkerbosch((4,3),(1), Φ)
  Bronkerbosch((4,3,1),Φ,Φ)
  Report (4,3,1) as one of the other Maximal Clique
- Bronkerbosch((4),(1,2,3,5,6),Φ)
  Bronkerbosch((4,2),(1,5),Φ)
  Bronkerbosch((4,2,5),Φ,Φ)
  Report (4,2,5) as an other Maximal Clique
- Bronkerbosch((4),(1,2,3,5,6), Φ)
  Bronkerbosch((4,6), Φ, Φ)
  Report (4,6) as the Maximal Clique

- This backtracking algorithm is a method for finding all the subsets in an undirected graph G.
- Given a graph G with 'V' vertices and 'E' edges, G = (V, E)
- Let us take an integer variable k.
- This algorithm is used in scientific and engineering applications.
- This algorithm is a Depth First Search algorithm.
- The algorithm for finding k-clique in an undirected graph is a NP-complete problem.

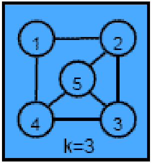

Figure: Graph

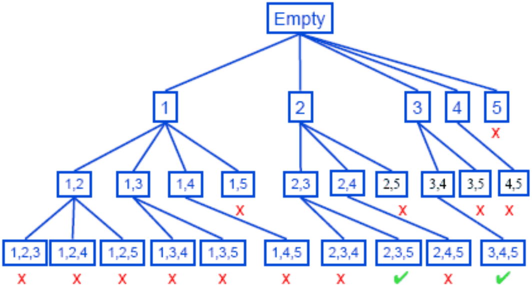

Figure: Backtracking

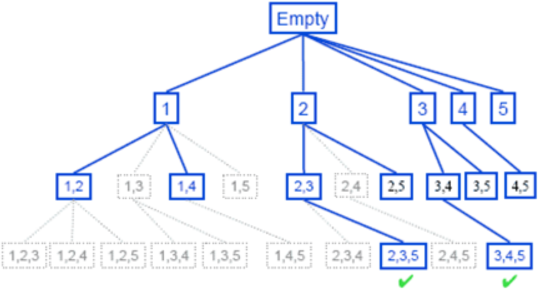

Figure: Backtracking

- List out all the possibilities in the sub graph and check for each and every edge.
- Check for a sub graph in which every node is connected to every other node.
- Check for all possible Cliques in the graphs.
- Check the size of clique whether it is equal to k or not.

- Any n-vertex graph has at most $3^{n/3}$ maximal cliques
- The worst-case running time of the Bron Kerbosch algorithm (with a pivot strategy that minimizes the number of recursive calls made at each step) is $O(3^{n/3})$
- This Backtracking algorithm runs in polynomial time if size of k is fixed. If k is varying then it is in exponencial time
- Running time of the algorithm is $O(nk)$, where k $= O(logn)$

- *Alon, N.Boppana,*
  {" The monotone circuit complexity of boolean functions"}
- *http : //en.wikipedia.org/wiki/Clique − problem.*
- *http : //www.ibluemojo.com/school/clique − algorithm.html*