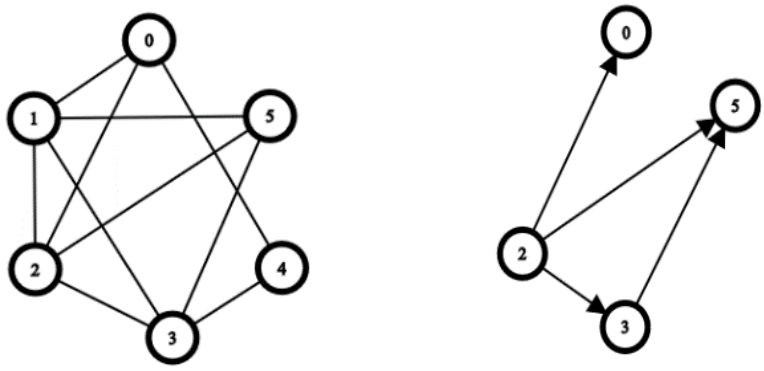
Algorithm Deployed:

- The following sequential algorithm is parallelized.

```
1: listing(k,G,∅)
2: function listing(l,G,C)
3:    if l = 2 then
4:       for each edge (u,v) of G do
5:          output k-clique C ∪ {u,v}
6:    else
7:       u1,. . . ,u|V (G)| ← nodes in G s.t. δ(ui) ≥ δ (ui+1)
8: for i = 1,. . . , |V (G)| do
9:    listing(l − 1,G[ΔG (ui)],C ∪ {ui})
10:   V (G) ← V (G) \ {ui}
```

Serial Algorithm

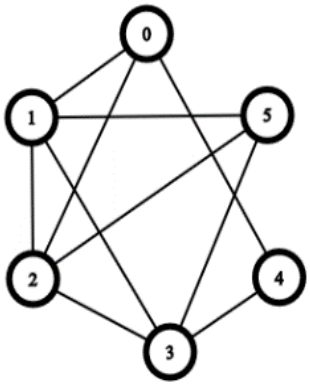
- The algorithm processes the nodes in non-increasing order of degree. For each node u, the algorithm computes the subgraph induced by its neighbors, and then it recurses on such a subgraph. When processing the nodes of a given subgraph, its nodes might have to be reordered in non-increasing order of degree. After processing a node u, u is deleted from the current graph so as to prevent that any clique containing u is listed more than once.
- Here G[ΔG(u)] is the induces subgraph, which is a graph that contains only those vertices and edges that are neighbors of current vertex. For e.g. G[ΔG(1)] is given below.



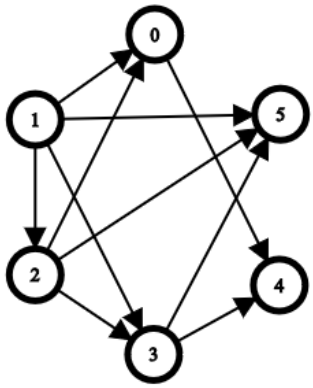
Given Graph Induced Subgraph of Node 1 i.e. G[ΔG(1)]

- To parallelize the above algorithm, we need to remove the serial dependency. Which is present in line 10.

- For this we'll convert the given graph into DAG, by taking the edges which comes from nodes with higher degree first, this is to avoid visiting the same vertex again. For eg after visiting node 1, it will not visit it further because there is no back edge to node 1.



Given Graph



DAG of given graph

- We can run every function call in parallel since every function call is independent.
- Below is the parallelized version of above serial algorithm.

Let η be a total ordering on the nodes of the input graph G
2: $\vec{G} \leftarrow$ directed version of G, where $v \rightarrow u$ if $\eta(v) < \eta(u)$
3: listing(k,$\vec{G}$,$\emptyset$)
4: function listing(l,$\vec{G}$,C)
5: if l = 2 then
6: for each edge (u,v) of $\vec{G}$ do
7: output k-clique C $\cup$ {u,v}
8: else
9: for each node u $\in$ V ($\vec{G}$) do
10: listing(l − 1,$\vec{G}$[$\Delta\vec{G}$(u)],C $\cup$ {u})

Reference:

[1] Maximilien Danisch, Oana Balalau, and Mauro Sozio. 2018. Listing k-cliques in Sparse Real-World Graphs