Identification and analysis of essential nodes via centrality measure and maximum cliques

Joona Lindell[1] and Fernando Dias[2]

[1]Department of Mathematics and Systems Analysis, Aalto University, Espoo, Finland, ✉ joona.lindell@aalto.fi
[2]Department of Mathematics and Systems Analysis, Aalto University, Espoo, Finland, ✉ fernando.dias@aalto.fi

1 Introduction

In a graph, an essential node (or vital node) is a node that belongs to a subset of high-importance nodes within a network. In this work, we refer to this problem as the subset of nodes from a graph, such that, without those nodes, the graph loses its integrity, creating disconnected components. While centrality methods are known for their flexibility and speed in handling moderately large instances, they lack optimality and remain open challenges. In this work, we propose a new pipeline that combines centrality measurements with clique and vertex cover problems to identify essential nodes. Through using the most valuable aspects of heuristics, such as runtime and simplicity with exact methods, we are aiming to obtain efficient methods capable of handling large instances, maintaining optimality.

2 Problem Formulation

In the context of this work, we considered a network $G = (V, E)$, where G is an undirected graph, composed of a set of nodes V and a set of undirected edges E. A *set of essential nodes* in this network represents a subset of nodes such that upon their removal, it results in a *disconnected graph.*

Definition 1 (Essential Nodes)**.** Essential nodes *in an undirected network corresponds to a set of nodes* $V' \subset V$ *and edges* $E' \subseteq E$*, such that, upon the resulting* $G' = (V', E')$ *is a graph with disconnected component.*

In previous research, most methods for identifying sets of essential nodes focused on centrality methods [2] and clique problems [1]. In this work, we start with a formulation to construct cliques and used as potential essential nodes. The construction proceeds by selecting a suitable starting node and then adding new vertices to the clique, subject to the condition that each new node is connected to all previously selected vertices. The selection process for the starting node (seed node) and the subsequent vertices, as described in Algorithm 1, uses centrality values to order the vertices.

Algorithm 1: Seeded centrality-order clique heuristic

Input: graph G, vertices in order of centrality values V, seed node s
Initialize $currClique = \{s\}$
Initialize $neighbourhood = l_G(s)$
for $n \in V$ **do**
  if $n \notin neighbourhood$ **then**
    continue
  if *n is connected to all vertices in currClique* **then**
    $currClique = currClique \cup \{n\}$
Return $maxClique$

Algorithm 1 runs a check on all possible vertices that could be part of a clique with a specific starting node. In practice, the vertices with the highest centrality are used as the seed node. An alternative algorithm is formulated by additionally recalculating centralities within the induced graph around the

seed node $G[l_G(s) \cup \{s\}]$. The original centrality values may contain a large amount of information from vertices not included in the set of possible vertices within the clique, see Algorithm 2.

Algorithm 2: SEEDED NEIGHBOURHOOD CENTRALITY-ORDER CLIQUE HEURISTIC

Input: graph G, centrality method C_M, seed node s
Initialize $currClique = \{s\}$
Initialize $neighbourhood = l_G(s)$
Calculate centralities for the graph $G[neighbourhood \cup \{s\}]$ using C_M.
Save vertices in the order of their new centrality values to V.
for $n \in \{s\} \backslash V$ **do**
  if *n is connected to all vertices in currClique* **then**
    $currClique = currClique \cup \{n\}$
Return *currClique*

3 Experimental analysis

In our preliminary results, we tested our setup using biological networks as inputs, analysing the runtime and comparing the output solution to an optimal solution obtained via an exact ILP formulation. For simplicity, we set a 5-minute time limit. In terms of efficiency, although our methods depend on different centrality measures, even with simpler ones, such as degree, they can still find an optimal solution, especially when starting with higher centrality values. Additionally, in some networks, depending on their density, the threshold for this value can be lower. In terms of runtime, our method is expected to be faster than traditional ILPs. The results show that even if the starting node has a lower centrality value, it is still possible to obtain an optimal clique size (or near optimal) within a time limit.

Figure 1: Clique sizes achieved for two inputs when all possible vertices are used as seeds for Algorithm 1 when utilising degree centrality ordering. The maximum clique size in the graphs is 250 (left) and 320 (right).

4 Conclusion and Future Research

In our work, we showed that there is potential in algorithms combining centrality measures and the maximum clique problem. Future research will focus on further testing with different centrality measures, such as eigenvalue and PageRank, and on combinations with different NP-hard problems (such as vertex and edge cover), as well as on applications with real data.

References

[1] Konstantinos A. Theofilatos, Christos M. Dimitrakopoulos, Athanasios K. Tsakalidis, Spyridon D. Likothanassis, Stergios T. Papadimitriou, and Seferina P. Mavroudi. Computational approaches for the prediction of protein-protein interactions: A survey. *Current Bioinformatics*, 6(4):398–414, 2011.

[2] Xiangmao Meng, Wenkai Li, Xiaoqing Peng, Yaohang Li, and Min Li. Protein interaction networks: centrality, modularity, dynamics, and applications. *Frontiers of Computer Science (print)*, 15, 01 2021.