Decision Support in Production, Logistics and Supply Chain

Arezoo Amiri Christof Brandstetter Tamara Ertl

March 21, 2024

1 Introduction

We only need to add edges with G.add edge(node1, node2, capacity) It automatically adds nodes

Pyvis does not add nodes automatically

net.toggle physics(True), sometimes makes sense to set to False, It makes the graph draggable

Generate C_l cut where $\gamma = 1$ and generate C_m where $\gamma = u$. Then calculate the bisection to obtain a new $\hat{\gamma}$. Then generate a cut with $\hat{\gamma}$ and do this until we find a cut that has less than B large edges.

2 Notation

- A cut is a partition $V = S \cup T$ of the nodes of G such that $s \in S$ and $t \in T$
- An arc $r \in E$ is in a cut $C = (S, T)$ if $\alpha(r) \in S$ and $\omega(r) \in T$
- Arcs having capacity u are called large arcs
- Arcs having capacity 1 are called small arcs
- $q(C) := \#(\text{large arcs in } C)$
- $p(C) := \#(\text{small arcs in } C)$

3 What's to do?

- Transform the digraph into a NFI graph
  - Create a s-node
  - Create a node for every edge in the digraph
  - Create edges from s to every *edge-node* i_x with capacity 2m

- Create a node for every node in the digraph
- Create edges from every *edge-node* to every *node-node* j_1 which is connected by the edges with capacity m
- Create a node t and connect the *node-nodes* with the t node by edges with capacity m
- where $m = |E|$

- Implement the bisection algorithm for NFI
  - Find minimum cut C
  - Identify the arcs $R \in C$ which are removed from the graph
  - C_m is optimal if it contains at least B large arcs with $val(C_m) = $ cap of arcs in cut − cap of removed arcs
  -
- Find in the NFI graph a strategy for NFI with budget $B = |E| - \binom{K}{2}$ that has value $K * m$ to get a clique of size K
- Transform it back to find the max-clique
- Use the bisection algorithm for NFI to find large cliques for the benchmark set
  - Suspect, that we have to do this for different K and raise the K's