A GRASP heuristic for the 1-Survivable Tree Star Problem

Louis Kurdyk, André Rossi, Sonia Toubaline

Université Paris Dauphine - PSL, CNRS, LAMSADE UMR 7243, 75016 Paris, France
firstname.lastname@dauphine.psl.eu

Keywords : *Tree Star problem, fault-tolerent network design, GRASP*

1 Introduction

Consider a complete mixed-weighted graph $G = (V, E \cup A)$. The Tree Star (TS) problem addresses the design of a network composed of two types of nodes: hubs and terminals. The backbone network is a spanning tree of the hubs using only edges from E. The remaining nodes are terminals, they constitute the tributary network, and each terminal is connected to a hub with an arc in A. The aim is to obtain this structure with minimum total cost, where edge costs follow a function $c : E \to \mathbb{R}^+$ and arc costs follow a function $d : A \to \mathbb{R}^+$. The term d_{ii} gives the cost of designating vertex $i \in V$ as a hub.

This problem has been presented by Lucena *et al.* in [1], and provides a mathematical formulation addressed with a Branch-and-cut algorithm. The problem's NP-hardness has been established in [2].

Given the need for fault-tolerant network structures, we introduce the 1-Survivable Tree Star (1-S-TS) problem, as an extension of the TS problem in which at most one hub can be down at any time, while maintaining connectivity. In this variant, we consider a set $\widetilde{V} \subseteq V$ of fallible vertices whose failure may compromise the graph connectivity. For example, if a hub h in the backbone network fails, the backbone may lose connectivity and break into components up to the cardinality of the neighborhood of h. Moreover, terminals connected to h become isolated from the rest of the graph, which results in loss of service, if the graph models a telecommunication network. The vertices in $V \setminus \widetilde{V}$ are regarded as secured since they are not subject to failures. This distinction between $\widetilde{V}$ and $V \setminus \widetilde{V}$ allows us to link network design problems that target tree-shaped topologies with those that aim for two-vertex connectivity, as we define the $\widetilde{V}$ fault tolerance:

Definition 1 *A network $N = (H, E' \subseteq E, A' \subseteq A)$, defined over the hubs of H, is $\widetilde{V}$-fault tolerant if for any $v \in \widetilde{V}$ the backbone network $B = (H \setminus \{v\}, E' \setminus \{\delta(v)\})$ is connected and for any terminal $t \in V \setminus H$ there is an arc $a \in A'$ from t to a hub h in $H \setminus v$. With for a given v in V : $\delta(v) = \{(i, v) \in E'\}$*

To address this problem, we develop both an integer linear programming formulation and a GRASP. The main emphasis of this paper is the latter, with the goal of extending and improving the solution proposed in [3] for the 2-Vertex-Connected-Star Problem.

2 Outline of the algorithm

The proposed Greedy Randomized Adaptive Search Procedure (GRASP) is divided into two main stages, a construction phase that generates an initial feasible solution and a local search phase that improves it. To avoid getting trapped in local optima, the second stage relies on a Variable Neighborhood Size (VNS) strategy based on four neighborhoods to explore the solution space more effectively.

2.1 Construction phase

The purpose of the construction phase is to build an initial backbone network containing a specified number of hubs. To explore a broader portion of the solution space, this phase uses randomization, which leads to multiple possible initial solutions. It begins by selecting three vertices uniformly at random and constructing the minimum cost $\widetilde{V}$-fault tolerant network using these vertices as hubs. Then, until the desired size is reached, a terminal t is chosen at random and introduced into the backbone network as a hub. To control the cost of this insertion, we develop an algorithm derived from the method presented by Bhandari in [4] to find pairs of minimum cost paths that do not share internal vertices in $\widetilde{V}$.

2.2 Local search phase

As mentioned earlier, the local search phase of the heuristic uses four neighborhoods, each corresponding to a specific operator. The first operator, *TerminalToHub*, takes a terminal t as input and generates a new solution by adding it as a hub in the backbone network. This insertion can be performed in two ways: either by connecting t to its nearest hub neighbors, or by splitting an existing edge in the backbone network. The first approach has a worst-case complexity of $O(|H|^5)$, where H is the set of hubs in the current solution. The second approach is faster, with a worst-case complexity of $O(|H|^3)$. However, the second method cannot be applied to all edges without risking violation of the connectivity requirements. To address this, we propose a characterization of the edges suitable for splitting, which allows us to guarantee the production of feasible solutions.

The second operator, *HubToTerminal*, removes a specified hub h from the backbone network and produces a solution in which h becomes a terminal in the tributary network. Since removing a hub can create multiple disconnected components, potentially up to the size of the vertex's neighborhood, we use the algorithm for finding pairs of minimum-cost paths that do not share internal vertices in $\widetilde{V}$ to ensure that the resulting solution remains valid and cost-efficient.

The *DisconnectReconnect* operator generates a new solution by reconnecting a given hub h differently within the backbone network. This can be viewed as a sequential application of the *HubToTerminal* and *TerminalToHub* operators. Introducing this operator helps avoid local optima where *HubToTerminal* alone would not yield improvements. Its correctness follows from the correctness of the two previous operators.

Finally, *ReoptimizeSubgraph* locally re-optimizes a subgraph of the current solution, either using an ILP model or a recursive call. These subgraphs may be either a tree or a 2-vertex-connected component. To identify subgraphs that can be rearranged without violating solution validity, we present a decomposition method based on the block decomposition of connected graphs.

The solutions obtained by this GRASP are compared with those obtained by solving the problem using an ILP model, allowing us to evaluate the heuristic's performance in terms of both computational speed and objective value.

References

[1] A. Lucena, L. Simonetti, A. Salles da Cunha, *The tree-star problem: A formulation and a branch-and-cut algorithm*, *Electronic Notes in Discrete Mathematics*, 52 (2016), 285–292.

[2] G.N. Frederickson, J. Ja'Ja', *Approximation algorithms for several graph augmentation problems*, *SIAM Journal on Computing*, 10 (1981), 270–283.

[3] R. Recoba, F. Robledo, P. Romero, O. Viera, *Two-node-connected star problem*, *International Transactions in Operational Research*, 25 (2018), 523–543.

[4] R. Bhandari, *Optimal physical diversity algorithms and survivable networks*, *Proceedings of the Second IEEE Symposium on Computers and Communications*, (1997), 433–441.