Minimum-Cost Bounded-Degree Spanning Tree

Girvar Patidar

Somya Bansal

Made with GAMMA

Problem Statement

Objective

Given a weighted undirected connected graph, find a spanning tree of minimum cost such that the degree of node v is no more than some specified bound.

Research Goals

- Remove reliance on Linear Programming (time-consuming, many constraints)
- Adapt local search techniques, and design a combinatorial approach to solve this problem.

Made with GAMMA

Local Search Algorithm: Unweighted Graphs

Problem Setup

Given $G = (V, E)$, find spanning tree T minimizing $\max_{v \in V} \deg_T(v)$

This is known as the **Minimum-Degree Spanning Tree problem**.

Computational Complexity

- Problem is **NP-hard**
- A spanning tree has $\Delta(T) = 2$ if and only if it is a **Hamiltonian path**
- Deciding if G has a Hamiltonian path is **NP-complete**

Exact solutions are computationally infeasible, motivating **approximation algorithms** and **local search heuristics**.

Made with GAMMA

Local Search Algorithm Design

Polynomial-time local search algorithm finds tree T with maximum degree at most $2 \cdot \mathrm{OPT} + \lceil \log_2 n \rceil$

Local Move Definition

- Let $d_T(u)$ = degree of node u in current tree T
- Consider all edges $(v, w) \notin T$ that create cycle C containing u
- Select edge where $\max(d_T(v), d_T(w)) \leq d_T(u) - 2$

Move Properties

- After local move: $d_{T'}(u) = d_T(u) - 1$, and $\max(d_{T'}(v), d_{T'}(w)) \leq d_{T'}(u)$
- Algorithm targets nodes with degree ≥ $\Delta(T) - \lceil \log_2 n \rceil$, and terminates if no local move is possible on any such node.

Made with GAMMA

OPT + 1 Local Search Framework

- Let D_k = all degree-k vertices in T
- Let F = edges incident to $D_k \cup D_{k-1}$
- Let C = components of T when F is removed

Phase-Based Execution

1. Algorithm operates in sequence of **phases**, each divided into **subphases**
2. Each phase removes all nodes of degree k from the tree
3. Each subphase targets a single degree-k node

Subphase :

- Identify degree-$(k-1)$ nodes in D_{k-1} for local moves
- Mark nodes as **reducible** via specific local moves
- Remove marked nodes from D_{k-1} and update F and C
- Eventually, we'll be able to perform a local move to reduce the degree of a node u in D_k

Made with GAMMA

Linear Programming Formulation

- Undirected graph $G = (V, E)$ with costs $c_e \geq 0$ for all $e \in E$
- Subset $W \subseteq V$ of degree-constrained vertices, and integer bounds $b_v \geq 1$ for all $v \in W$

LP Formulation

Minimize: $\sum_{e \in E} c_e x_e$

Subject to:

- $\sum_{e \in E} x_e = |V| - 1$ (spanning tree constraint)
- $\sum_{e \in E(S)} x_e \leq |S| - 1, \forall S \subseteq V, |S| \geq 2$ (acyclicity)
- $\sum_{e \in \delta(v)} x_e \leq b_v, \forall v \in W$ (degree bounds)
- $x_e \geq 0, \forall e \in E$ (non-negativity)

Made with GAMMA

LP-Based Deterministic Rounding

1. Solve LP relaxation and obtain basic optimal solution x
2. Work with support $E(x) = \{e : x_e > 0\}$

Iterative Reduction Process

While $|V| > 1$:

1. **Remove zero-edges** from consideration
2. **If** vertex v has exactly one incident edge (u, v) in $E(x)$:
   - Add (u, v) to solution F
   - Remove v and decrement b_u if $u \in W$
3. **Else if** $v \in W$ has ≤3 incident edges in $E(x)$: remove v from W

Returns spanning tree F with:

- Cost ≤ LP optimal value
- Each $v \in W$ satisfies $\deg_F(v) \leq b_v + 2$

Made with GAMMA

Refinement to +1 Violation Bound

Any basic feasible solution x (when $W \neq \emptyset$) contains some $v \in W$ with at most $b_v + 1$ edges in $E(x)$.

Improved Algorithm

1. Repeatedly solve LP relaxation
2. Remove vertex $v \in W$ guaranteed by lemma (≤ $b_v + 1$ support edges)
3. When W becomes empty, compute ordinary minimum-cost spanning tree on remaining support

Produces spanning tree with:

- Cost ≤ LP optimal value
- Each $v \in W$ satisfies $\deg_F(v) \leq b_v + 1$

Made with GAMMA

Flow-Based Constraint Reduction

Motivation

Replace exponential number of cycle constraints with polynomial-sized flow formulation

Extended Variable Set

- Original edge variables: x_e for each undirected edge $e = (u, v)$
- Flow variables: $f_{uv}^a \geq 0$ for each ordered arc (u, v) and commodity $a \in V \setminus \{r\}$

Flow Constraints

Capacity constraints:

$$\forall \{u, v\} \in E, \forall a \in V \setminus \{r\} : f_{uv}^a \leq x_{uv}, f_{vu}^a \leq x_{uv}$$

Per-commodity supply/demand:

$$\sum_{w:(v,w)\in A} f_{vw}^a - \sum_{w:(w,v)\in A} f_{wv}^a = \begin{cases} 1 & \text{if } v = a \text{ (source)} \\ -1 & \text{if } v = r \text{ (sink/root)} \\ 0 & \text{otherwise} \end{cases}$$

Made with GAMMA

What we have done so far..

- Studied minimum-cost bounded-degree spanning tree problem.
- Explored local search algorithms for unweighted graphs.
- Studied existing LP-based approaches with rounding techniques.
- Implemented and tested the LP algorithm that, given a graph and degree bounds, returns the optimal bounded-degree MST.
- Investigated constraint reduction using flow formulations.

What we plan to do next..

- Extend local search methods to weighted graphs.
- Explore new combinatorial heuristics to reduce LP dependence.

Made with GAMMA