Isolation Branching for k-terminal cut

Mark Velednitsky, Dorit S. Hochbaum

University of California, Berkeley

We developed a custom branch-and-bound algorithm for the `k-terminal cut` problem which does not rely on an IP formulation. Our algorithm employs "isolating cuts." We compared the performance of Isolation Branching to that of Gurobi solving the strongest known IP formulation. Isolation Branching runs an order of magnitude faster on instances with hundreds of thousands of edges. The complexity of Isolation Branching is fixed-parameter tractable with respect to the size of the optimal solution.

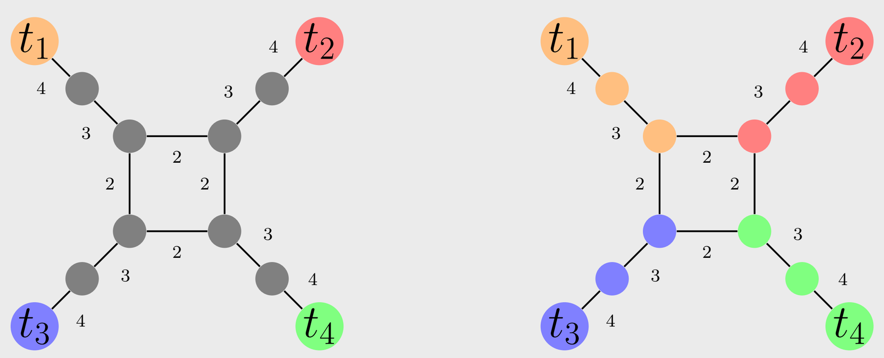
k-terminal cut

Input Weighted graph $G = (V, E)$ and terminals $t_1, \dots, t_k$.

Output Assign all the vertices to terminals: $(T_1, \dots, T_k)$.

Objective Minimize the total weight between terminal sets:

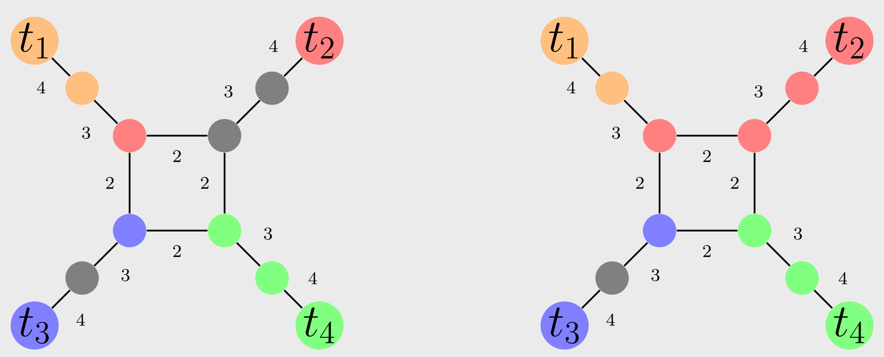
$$\min \frac{1}{2} \sum_{i=1}^{k} w(T_i, V \setminus T_i).$$



Input and output for an instance of the `k-terminal cut` problem.

Isolating Cuts

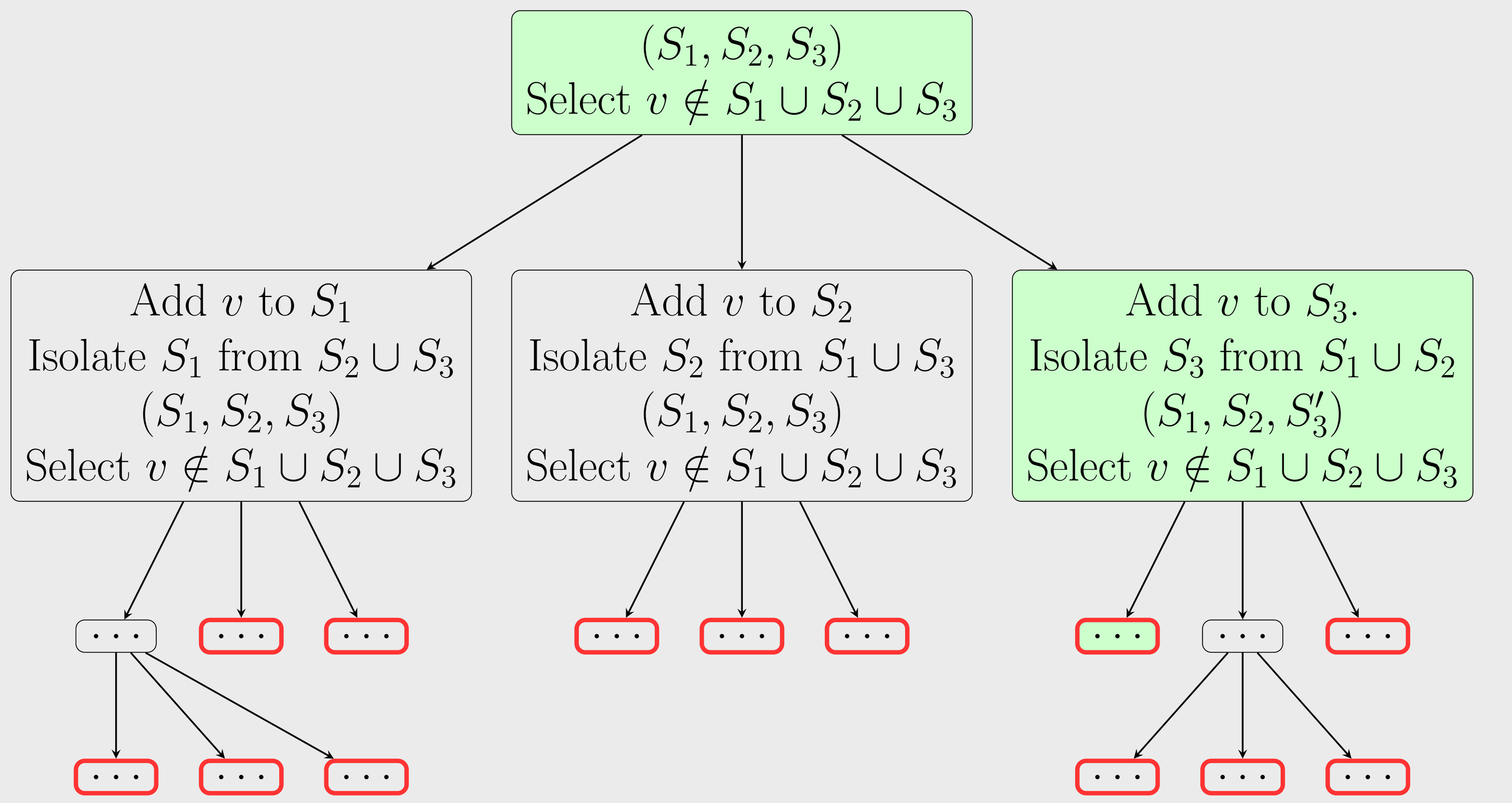
Given a collection of subsets of vertices $(S_1, \dots, S_k)$, an **isolating cut** for S_i is a minimum cut separating S_i from all the other S_j.
In Isolation Branching, vertices in the source set of the cut are added to S_i.



Before and after an isolating cut separating S_2 from $S_1 \cup S_3 \cup S_4$.
Vertices in the source set of the cut are added to S_2.

Isolation Branching

At each node, we store sets $(S_1, \dots, S_k)$.
We pick a vertex $v \notin S_1 \cup \dots \cup S_k$ to branch on.
In each child node, we add v to a set, then grow the set with an isolating cut.



Correctness

A collection of sets $(S_1, \dots, S_k)$ has the containment property if there is an optimal cut $(T_1, \dots, T_k)$ such that $S_i \subseteq T_i$ for all i.

The optimal solution will appear at a leaf of the enumeration tree because:

- The root node has the containment property.
- If a node has the containment property, then at least one child has it.
- A leaf node with the containment property is the optimal solution.

Bounding

At each node, let $L(S_1, \dots, S_k)$ be half of the sum of the weights of the k isolating cuts. The function L operates as a tight lower bound:

Lower Bound: The value of L increases from parent node to child node.

Best Bound: The smallest L among unexplored nodes is the best bound.

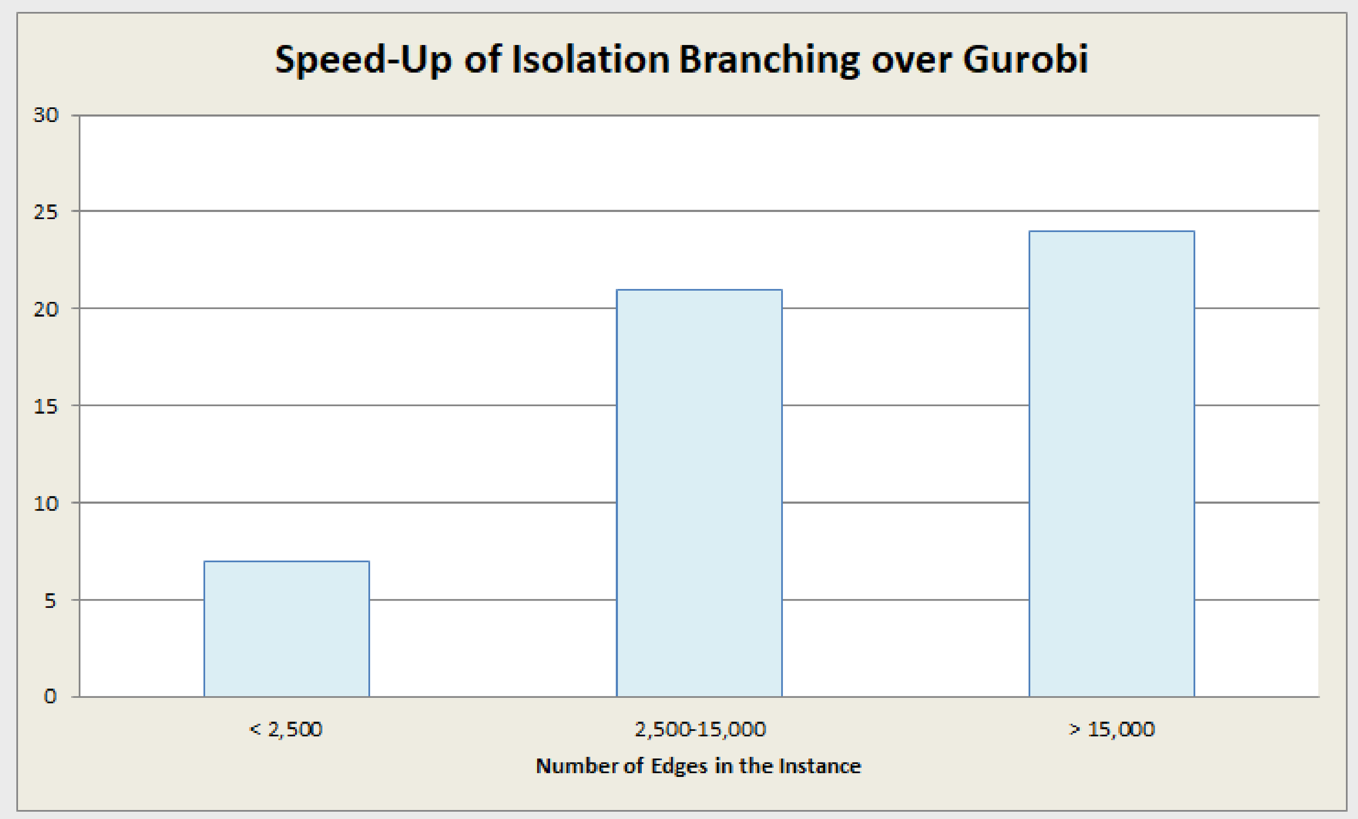
Tightness: At a leaf, L equals the value of the feasible solution.

We use the best-bound strategy. The first leaf we encounter is optimal: it contains a feasible solution whose value equals the best bound.

Empirical Results

Isolation Branching provides a median speed-up over Gurobi of 18x.

In instances with more than $15,000$ edges, the median speed-up is 24x.



https://github.com/marvel2010/k-terminal-cut