Steiner Tree Approximations in Graphs and Hypergraphs

Miklos Molnar *

Posted Date: 20 February 2026

doi: 10.20944/preprints202602.1350.v1

Keywords: graph theory; networks; multicast diffusion; combinatorial optimization; Steiner tree; hypergraphs; heuristic algorithms

 doi:10.20944/preprints202602.1350.v1

Article

Steiner Tree Approximations in Graphs and Hypergraphs

Miklos Molnar

LIRMM, University of Montpellier, CNRS, 161 rue Ada, 34095 Montpellier Cedex 5 France; molnar@lirmm.fr

Abstract

The construction of partial minimum spanning trees being NP-hard, several heuristic algorithms have already been formulated. Many of these heuristics (such as Kruskal's) use shortest paths to connect the components of the tree. In this work, we present an approximate construction algorithm for the minimum Steiner tree (the optimal tree for diffusion multicast). This construction is based on graph-related structures more advantageous than shortest paths. The algorithm uses connections like simple Steiner trees if necessary. These simple trees can be represented by hyperedges. A hyper metric closure can also be used.

Keywords: graph theory; networks; multicast diffusion; combinatorial optimization; Steiner tree; hypergraphs; heuristic algorithms

1. Introduction and Related Work

The Steiner problem is one of the most studied problems in combinatorial optimization. In its version formulated in Euclidean space, the minimum cost/length connection of a set of points (places, cities) is of significant interest in several logistical decisions. In the design of microcircuits and electronic boards, optimal connections of elements and pins involve the rectilinear Steiner tree and Steiner forest problems. A set of constraints can be given by a graph where edges indicate possible connections. The model can correspond to a network topology in which a multicast group must be created for communication. The connection must be established with minimal cost. In our study, this is the version analyzed.

Let the network be represented by a non oriented, connected graph $G = (V, E)$ where V and E denote the sets of nodes and edges respectively. Let us suppose that positive lengths (weights) are associated with the edges. A multicast group $M \subseteq V$ of nodes is given in the graph G. Our objective is the construction of the cost optimal connection for the group members. It is well known, that it is a minimum cost partial spanning tree (Steiner tree) [1]. If $M = V$, the solution is a Minimum Spanning Tree (MST) and its construction is very easy (*cf.* the algorithm proposed by Kruskal [2] and also by Prim [3]). On the contrary, the Steiner problem is NP-hard.

Note that the most important definitions can be found at the en of the document in Appendix.

There are simple heuristics to built approximate partial minimum spanning trees using polynomial execution time. The algorithm of Kruskal can be applied using the shortest paths between the group members. The solution of Takahashi and Matsuyama [4] is based on the Prim's algorithm but using also the shortest paths. The construction of Kou, Markovsky et Berman presents the problem directly in the metric closure [5]. These algorithms guarantee 2-approximations of the minimum Steiner trees. An 11/6 - approximation is proposed by the greedy algorithm of Zelikovsky in [6]. This proposal is also based on the metrical closure. To improve the simple MST algorithm [5], small Steiner trees for triplets of members are examined. Let $MST(M)$ be the MST of the group M in the metric closure. For each triplet z, the minimum Steiner tree $T(z)$ is computed in the original graph. The central node $c(z)$ of $T(z)$ is then considered as a new node to cover if $gain(z) = mst(M) - mst(M') - d(z)$ is positive and maximal.

 doi:10.20944/preprints202602.1350.v1

In this function, $mst(M)$ corresponds to the length of $MST(M)$ and $mst(M')$ is the length of the MST in the modified metric closure after the contraction of the nodes in z. $d(z)$ is the length of the minimal Steiner tree covering the triplet z. Each triplets of M are examined and the central nodes of the appropriate small Steiner trees are added to the metric closure (*cf.* figure 1). Finally, the approximation of the Steiner problem is computed in the augmented graph. THe result is a 3-restricted tree.

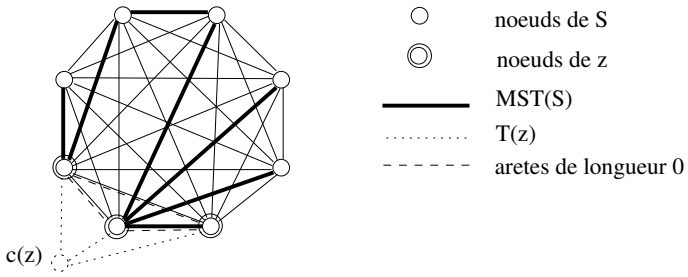

Figure 1. Replacement a part of an MST by adding a small Steiner tree

In a later study, Zelikovsky provided a framework for greedy algorithms and introduced the relative greedy heuristic (RGH) using a special selection function [7]. The proposed framework of algorithms works on a class K of partial Steiner trees and selects trees with a generalized objective function $f(T)$.

Efficient pre-treatment methods have been proposed by Karpinski and Zelikovsky in [8]. The authors attempt to put into perspective the gains and losses of the potential components of a partial spanning tree. The described preprocessing phase aims to find the most advantageous full components from a relative gain perspective. Once such trees are found, their S-nodes (called also Steiner points) are added to the set of nodes to be covered.

More general greedy algorithms have been proposed. Du, Zhang and Feng [9] proposed an algorithm for the Euclidean Steiner problem by using k-tuples. The version applied for the graph related problem can be found in [10].

Note that, it is always possible to progressively create a tree spanning M by adding nodes of M applying a greedy strategy.

Let be $\tau \in M$ be a new node and let $T(\tau)$ be a Steiner tree connecting τ to the MST by k nodes of the MST (*cf.* Figure 2). Adding $T(\tau)$ to $MST(M')$, $k-1$ cycles are created. $k-1$ edges of the metric closure can be revised and a new MST can be computed.

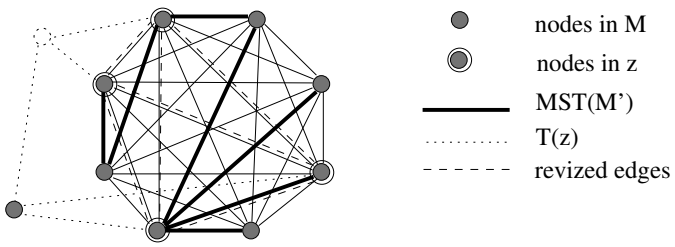

Figure 2. Revision of some edges in the metric closure

Berman and Ramaiyer proposed a solution in two phases [11]. At a first time, their algorithm enumerate the k-tuples with positive gain and a second phase allows the creation of the approximate tree.

One of the problems of greedy heuristics is the following. If the algorithm chooses a (k-tree) component based on the maximum gain (or relative gain) during an iteration, the contraction of the k-tuple can limit subsequent choices and may exclude components that are more suitable for the final solution. It is this drawback that the Berman-Ramaiyer algorithm and the Karpinski-Zelikovsky pretreatment attempt to compensate for, respectively, by delaying the selection of components of the final solution or by proposing more S-nodes of the solution. Robins and Zelikovsky propose another

 doi:10.20944/preprints202602.1350.v1

solution to minimize these "losses" by implementing a different greedy heuristic. The algorithm modifies a terminal spanning tree T (which is initially the MST in the metric closure) by adding into T loss-contracted full components greedily chosen from G, based on their "relative cost savings" [12] .

An improved approximation factor of a special optimal Steiner tree can be found in [13]. The solution is an LP-based approximation algorithm based on the iterative randomized rounding technique. k-restricted Steiner trees are investigated and a directed-component cut relaxation for the problem. The solution is of cost at most ln(4) times the cost of the optimal k-restricted Steiner tree.

A large online compendium on the approximability of the Steiner Tree related optimization problems can be found at
https://theory.cs.uni-bonn.de/info5/steinerkompendium/.

An in-depth study of the Steiner problem in graphs, with a description of the most important exact solutions, heuristics, and approximations, is presented in [14].

2. Hypergraphs and Steiner Trees

To solve the full Steiner tree concatenation problem in Euclidean Steiner problem, Warme proposed a hypergraph-based model [15,16]. That is, to create an optimal Steiner tree in the space from discovered full components on the set of goints to connect, the full component can be considered as hyper-edges. Then the final Steiner tree is computed by solving the Minimum Spanning Tree problem in a hypergraph. To find the Minimum Spanning Tree in a hypergraph is NP-hard, and Warme gives an elegant proof.

Note that any Steiner tree in graphs can be partitioned into a set of edge-disjoint but connected full components by splitting at internal terminals. Each full component can correspond to a hyper-edge and the model gives a hypergraph presenting the Steiner tree problem.

In [17] the author indicate that the problem is related to the MST problem in weighted 3-uniform hypergraphs and give a fully polynomial randomized approximation scheme.

A Steiner tree representation of the hyperedges is considered as the edge representation of a hypergraph in [18]. To solve the Steiner Tree problem, the metric closure (the distance graph) can be used as it is also indicated in the chapter [19]. This last chapter gives an analysis and important relations between the Steiner Tree problem in graphs and the Minimum Spanning Tree problem in hypergraphs. A branch-and-cut method based on the linear relaxation of an integer program of the problem has been proposed.

Paper [20] compares several relaxations of the problem based on hypergraphs as well as relaxations based on the formulation of the Steiner problem in graphs. It indicates that the union of trees linked by the hyperedges is a graph, and that the concatenation must be reduced by solving the classical Steiner problem in this graph.

Small Steiner trees corresponding to hyperedges can be used as connections between trees, and also to connect members to trees covering a group. The first propositions can be found in [21,22]. These concepts will be developed in the next of this study.

3. Bases of the Proposed Solutions

The following property of optimal Steiner trees allows approximate trees to be constructed by concatenating smaller components. These components are not obligatory full components.

Suboptimality of Steiner trees

Let $T = (V_T, E, T)$ be a minimum Steiner tree of the node set M. Let $ST = (V_S, E, S)$ be a subtree of T delimited by the leaves $L_S \subset V_T$. Let $M_S = V_S \cap M$ be the set of nodes in ST belonging to M. ST is a minimum Steiner tree of the set $L_S \cup M_S$.

The proof is trivial. Let us suppose that ST is not minimal to cover $L_S \cup M_S$, and another minimal tree ST' exists. T can not be minimal (replacing ST by ST' a smaller Steiner tree is obtained).

This property can be the base of the construction of Steiner trees in the generalized metric closure. The problem is that the good decomposition of the optimal Steiner tree into smaller optimal Steiner

 doi:10.20944/preprints202602.1350.v1

trees is not known. The concatenation of optimal small trees does not result in an optimal tree. For instance, the set sof shortest paths (small Steiner trees) covering M does not give an optimum. Figures 3 illustrates the case. The minimum Steiner tree of $\{a, b, c, d\}$ is the tree $T(a, b, f, g, c, d)$ with length 32, and this tree does not contains shortest paths between the members nor the minimum Steiner tree $T(a, b, e, d)$.

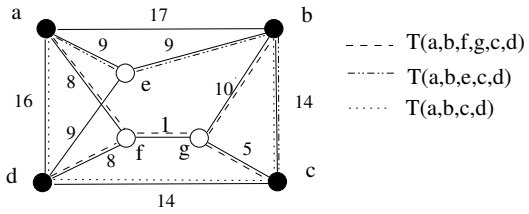

Figure 3. Different trees spanning M

Greedy algorithms are known to create approximate Stienr trees by merging/ adding the nodes in M s toa tree. The heuristic of Takahashi and Matsuyama and the algorthm of Kruskal are good examples: members of M are added or connected by shortest paths.

In previous studies, we proposed the modification of these algorithm [21,22]. Instead of shortest paths, small Steiner trees can be used as connections.

3.1. Connections and Distances

The connection of trees by trees permits the revision of the "distance" between the trees.

Figure 5 illustrates a simple case: a group M of nodes is given by the filled nodes, Two separated trees, T_1 and T_2, cover two subsets of M. Our goal is the "optimal" unification of the two trees to obtain a common partial spanning tree of minimal length, if possible.

To address the problem of optimally connecting two trees to obtain a spanning tree, different connections / " distances" between the trees can be defined.

Optimal connection of a node n to a Steiner tree T [22]

Let T be a Steiner tree of the group M. Let $C(l)$ be a Steiner tree with l leaves having n as leaf and let the other $l-1$ leaves of $C(l)$ be nodes in T such that $C(l)$ and T are edge disjoint. Let $G(T, C(l))$ the union of T and $C(l)$. In general, $G(T, C(l))$ is not a tree, and can contain cycles. Let $D(T, C(l))$ be the set of edges with maximal length that can be deleted from T without loss of connectivity of $M \cup \{n\}$ and let $T(T, C(l))$ the tree after removing $D(T, C(l))$ from T.

The tree $C^*(l)$ is an optimal connection of l leaves between n and T if the length of $T(T, C^*(l))$ is minimal. We call this length the $(l-2)$-distance of n from T (in this way, the usual distance is the 0-distance).

The optimal connection of n and T is obtained with $l*$ leaves

$$lengthT(T, C^*(l^*)) = \min_l lengthT(T, C^*(l))$$

The *complexity* of the computation of the optimal connection depends on the size of the concerned Steiner tree. If the tree has n_T nodes, the $l-1$ leaves of $C(l)$ can be placed in $\binom{n_T}{l-1}$ different ways on the tree.

This kind of connections and distances can be defined to connect to Steiner trees.

Optimal connection between two Steiner trees

Here also, the objective is to create a unique Steiner tree, when two distinct Steiner trees are given to cover two subsets of the nodes in M. Let T_1 and T_2 two Steiner tree Steiner trees covering M_1 and M_2 respectively; $M = M_1 \cup M_2$. Let $C(l)$ be a Steiner tree with l leaves such that its leaves are in T_1 and T_2. Let L_1 and t_2 the set of leaves of $C(l)$in T_1 and T_2 respectively, and $L_1 \neq \emptyset$, and $L_2 \neq \emptyset$. Moreover $C(l)$, T_1 and T_2 are edge disjoint.

Let $G(T_1, T_2, C(l))$ the union of T_1, T_2 and $C(l)$.

 doi:10.20944/preprints202602.1350.v1

Let $D(T_i, C(l)), i = 1,2$ be the set of edges with maximal length that can be deleted from T_i without loss of connectivity between the members of M_i. let $T(T_1, T_2, C(l))$ the tree after removing $D(T_i, C(l))$ from $T_i, i = 1,2$.

The tree $C^*(l)$ is an optimal connection of l leaves between T_1 and T_2 if the length of $T(T_1, T_2, C^*(l))$ is minimal.

We call this length the **(l-2)-distance** between two Steiner tree. Following this concept, the usual distance between two trees is the 0-distance. The 1-distance is obtained using a 3-tree as connection, etc. These connections are illustrated by Figure 5.

The absolute optimal connection is not limited to a specific number of leaves. This connection of T_1 and T_2 is obtained with l^* leaves

$$lengthT(T_1, T_2, C^*(l^*)) = \min_l lengthT(T_1, T_2, C^*(l))$$

The *complexity* of the computation of the optimal connection between the Steiner trees depends on the size of the trees. Let us suppose that the trees have n_{T_1} and n_{T_2} nodes. The l leaves of $C(l)$ can be placed in $\binom{n_{T_1}+n_{T_2}}{l}$ different ways on the tree. There is no connection, when all leaves are in the same tree. This is produced in $\binom{n_{T_1}}{l} + \binom{n_{T_2}}{l}$ cases (supposing that l is less than n_{T_1} and n_{T_2}). The number of different connections is $\binom{n_{T_1}+n_{T_2}}{l} - (\binom{n_{T_1}}{l} + \binom{n_{T_2}}{l})$

For each connection, a set of edges can be deleted from the trees. Trivially, the sets $D(T_i, C(l))$ of edges that can be deleted from the mentioned Steiner trees form forests.

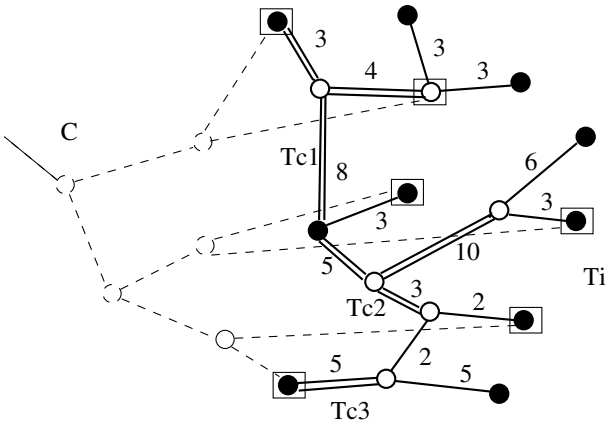

Figure 4. Forest to remove from a tree T after adding another tree C

Figure 4 shows a Steiner tree T covering the nodes of a set M. This tree should be connected to other members via a tree C and the result should be a tree. Trivially, to avoid cycles, a part of T should be removed. This part is in the subtree delimited by the leaves of C and corresponds to a forest. The following lemma gives a criterion to select the edges to remove.

Lemma 1. *Let $T_i = (V_i, E_i)$ a Steiner tree for $M_i \subset V_i$, and C a connected tree. $\tau \subset V_i$ is the leaves of C in T_i. Let A be a tree with maximum cost such that $T_i - A$ contains at least one node from τ (a leaf of C). If T_i is connected to another elements via C, there is a forest with maximum cost that can be removed from T_i and contains A.*

 doi:10.20944/preprints202602.1350.v1

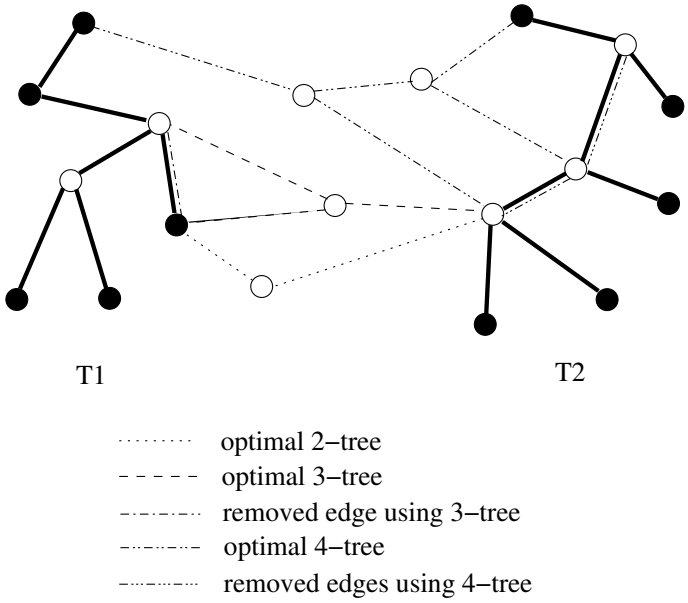

Figure 5. Connections with k-trees and related distances between two Steiner trees

4. Hypergraph Model and Metric Closure

To illustrate and handle Steiner trees, a hypergraph based model can be used. The model is more detailed in [23].

This model can be very usefull, if some parts, sub-trees of a desired Steiner tree of a group M are known or precomputed. The objective is to cover the set of group nodes M by a unique tree using the limited size known components.

Hyperedge A component (a tree) can be represented by a hyperedge. Let T_s be a known part (a subtree) of a Steiner tree. This component can be represented by a hyperedge e_s and let the cost ofT_s considered as the cost $c(e_S)$ of e_S.

Coverage A tree T is covered by a set of hyperedges, iff all of the nodes of the tree belong to at most one hyperedge.

Connectivity Two hyperedges are connected, if they share at least one common node.

To ensure connectivity between the nodes using a set of hyperedges, the hyperedges must be connected; there must be a path connecting each node to the others via the set of hyperedges.

In special cases, the coverage of a tree can be a chain of hyperedges (a path) or a hyper-tree.

Cost of a coverage The cost of a coverage is the sum of the costs of the hyperedges that compose it.

Lemma 1. *Let $S(T) = \{e_k, k = 1, ..., K\}$ be a set of hyperedges covering a tree T. The cost of $S(T)$ is at least the cost of T.*

$$c(S(T)) = \sum_{k=1,..,K} c(e_k) \geq c(T)$$

Proof. Each hyperedge corresponds to a sub-tree of T. All of the nodes in T are covered. Suppose that the cost of T is greater than the sum of the costs of the hyperedges. In this case, there is at least an edge of T which is not included in any hyperedge. The connection between the nodes of T by the hyperedges is not complete. The contradiction is trivial. □

Often, to find good coverage for sets of nodes, metric closure graphs can be used Usually, the metric closure contains the distances between the nodes. The distance of two nodes is the ”cost” of the shortest path between them. Trivially, a shortest path is a hyperedge of size 2 (a 2-tree).

The metric closure can be generalized on the basis of larger (but limited) size hyperedges.

 doi:10.20944/preprints202602.1350.v1

Generalized metric closure, k-limited metrical closure Let M be a set of nodes in a connected graph. Let $k < m = |M|$. The set of hyperedges of size limited by k forms a k-limited metric closure. In this hypergraph, the hyper-edges are k-limited; the correspondig trees are at most k-trees.

The generalized metric closure hypergraph contains the usual metric closure (the shortest paths, but also all of the hyperedges based on triplets of nodes, etc. Figure 6 illustrates some hyperedges in the generalized metric closure of a small group.

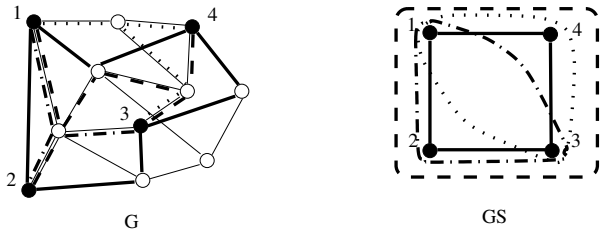

Figure 6. Some hyperedges in a generalized 3-limited metric closure

We suppose that the intersection of the hyperedges belonging to a coverage of a tree T contains *only one node per peer*. To minimize the redundancies between the trees associated with the hyperedges, this condition is necessary. Let us suppose that the intersection of two hyperedges contains several nodes. In this case, sub-graphs delimited by this set of nodes are present in the trees and result useless cost for a good coverage.

The optimal coverage using k-limited components can be computed as follows.

To find the coverage with minimal cost for an arbitrary value $2 < k < m$, different exact algorithms can be found. The following procedure has two main steps:

1. For the set of nodes M, construct the k-limited hyper metric closure HMC. For the integer values i from 2 to k, compute all i-tuples. Associate the cost of corresponding sub-trees with the hyperedges. Remember that, for a given value i, there are $\binom{m}{i}$ tuples to create.
2. Find the coverage of M with minimal cost using the best combination of the hyperedges. To do this, all sets of connected hyperedges covering M must be enumerated, and the best cost set must be selected.

This procedure does not allow us to find the minimum cost Steiner tree, but the best tree that can be built using k-limited components. The steps of this algorithm to find this good coverage are very expensive. A greedy algorithm without guarantees regarding costs can be created as follows:

1. Starting from an arbitrary node in M, a first subset of M is determined: the $k-1$ nearest nodes in M are added to the first one to form a set $S1$. A first hyperedge e_{S1} is then created on the set $S1$ and the Steiner tree covering $S1$ is associated with it.
2. At each step of the suite, the tree corresponding to the last hyperedge is examined. To create a new connected hyperedge, a leaf from this hyperedge is selected, and the $k-1$ nearest not yet covered nodes of M are added to this one to form the next hyperedge. Since one leaf of the previous hyperedge belongs to the new hyperedge, a chain is created. The construction ends when all nodes of M are covered.

Note that the best cost solution is not necessarily a chain, but can also be a tree. Furthermore, regarding chains, the choice of the start point and the subsequent continuations also influence the cost. Greedy decisions do not guarantee an optimal cost chain either. As indicated, the problem is NP-hard.

A performance analysis of the k-limited approximations of Steiner trees based on the hyper metric closure can also be found in [23].

5. Conclusions

In this work, an overview of the most known heuristics to solve the Steiner problem in graphs is given. A good part of the algorithms is based on the shortest paths. In some cases, the constructed

 doi:10.20944/preprints202602.1350.v1

trees are improved by adding small components, partial spanning trees. In these cases, the revision of the edge set is necessary, some edges can be removed.

Recently, hypergraph based construction have been proposed. Partial spanning trees can be represented by hyperedges. They can be concatenated using greedy algorithms, or a hyper metric closure can be built and approximate solution can be foind using it.

The construction of efficient heuristics for the aformentioned NP-hard optimization is a challenging goal.

Appendix A. Definitions, Related Notions

In this section known and often referenced concepts and definitions are grouped.

Definition 1 (Steiner nodes or S-noeuds). *A Steiner tree T spanning a given nodes set M can contain nodes in V which are not in M. Often, they are called as Steiner nodes (or Steiner points). In our discussion, we refer to Steiner nodes or S-nodes for this type of nodes when their degree in the Steiner tree is greater than 2.*

Definition 2 (Metrical closure). *Let $M \subseteq V$ be a set of nodes in G. Let $G_M = (M, D)$ be a complete graph on M such that the length of an edge $d(a, b) \in D$ corresponds to the length of the shortest path between a and b in G. G_M is the metrical closure of M (or its distance graph).*

Definition 3 (Full Steiner tree). *[11]*

A Steiner tree T spanning M is full, if the internal nodes of T are not in M.

Definition 4 (k-tree). *[11]*

A k-tree (k-restricted or k-limited Steiner tree) is a full Steiner tree with at most k leaves.

Consequently, a 3-tree is a star.

Definition 5 (Partition of a Steiner tree into a set of full components). *[11]*

A Steiner tree of M can always be partitioned into a set of components, which are adjacent, edge-disjoint full Steiner trees sharing leaves in M (cf. figure A1).

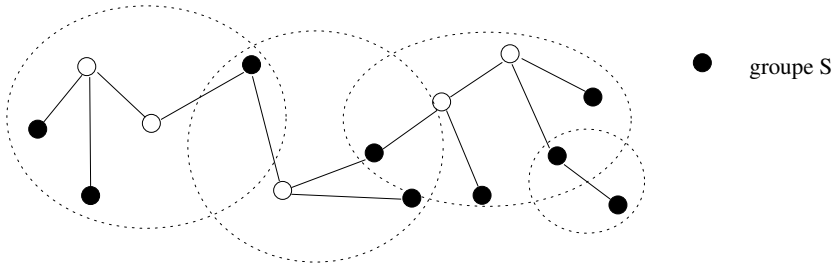

Figure A1. Partition into full components

Definition 6 (k-restricted Steiner tree). *[11]*

A Steiner tree is k-restricted, if the components in its partition are k-trees.

In a connected graph and for a group M, it is always possible to create k-restricted Steiner trees covering M, but without a guarantee of optimality. For instance, a 2-restricted Steiner tree is a tree that connects the group members by the shortest paths.

Definition 7 (M-covered forest in a k-tree). *[7]*

Let K be a k-tree of M and let $V(K)$ be the set of branching nodes in K. A forest $F \subseteq K$ is called M-covered, if $\forall v \in V(K)$, there is path in F, which links v with an element of M (cf. figure A2).

 doi:10.20944/preprints202602.1350.v1

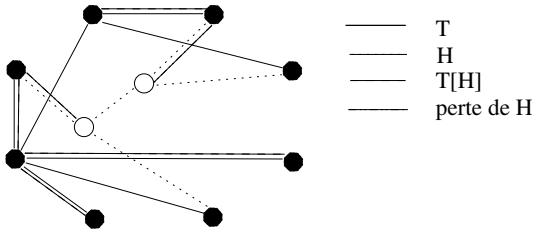

Figure A3. Saving and loss related to a component H

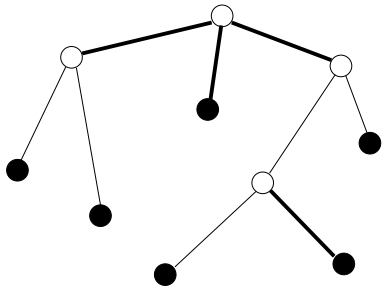

Figure A2. M-covered forest in a k-tree

Definition 8 (Minimum M-covered forest in a k-tree)**.** *[7]*

In a k-tree, several M-covered forests can exist. One of them is with minimum cost.

Let K be a tree. Loss(K) is a minimum-cost subgraph of K containing a path from each Steiner point of K to one of the terminals of K.

Definition 9 (Terminal Steiner tree)**.** *[12]*

A Steiner tree that contains only terminal nodes (nodes in M) is a terminal Steiner tree.

Note : In a complete graph (and so in the metric closure of the Steiner problem), there are always terminal Steiner trees.

Definition 10 (Gain related to a component H)**.** *[12]*

Let T be a terminal Steiner tree (of M) and let H be a component connected to a subset of nodes in T. The gain of H related to T is:

$$gain_T(H) = d(R_H) - d(H)$$

where R_H is the set of edges (a forest) of maximal length that can be removed from T preserving a Stiener tree after the union of H and T (cf. figure A3).

Definition 11 (Saving of H)**.** *As previously, let T be a terminal Steiner tree and let H be a component connected to a subset of nodes in T.*

The saving of H related to T is:

$$saving_T(H) = d(T) - d(H) - d(T[H])$$

ou $T[H]$ is the forest with minimal length spanning the terminals in T that are not covered by H (cf. figure A3).

If H is a full component, $T[H]$ is a tree and corresponds to the saving.

Definition 12 (Loss of a component)**.** *[12]*

The loss (indicated by $Loss(H)$) of a component H is the length of the minimal length forest composed from edges linked the S-nodes of H to the terminal nodes.

 doi:10.20944/preprints202602.1350.v1

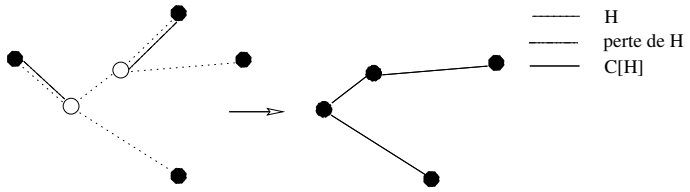

Figure A4. Loss contracted full component

Definition 13 (Gain α-relative). *[8]*

La gain α-relative of a full component H is defined as follows :

$$gain_k(H, \alpha) = d(H) - \alpha \cdot loss(H)$$

Definition 14 (Loss contracted Full component). *[12]*

Let H be a full component. Each tree in $Loss(H)$ after contraction correspond to a composed node that considered as a terminal node.

The corresponding loss contracted full component $C[H]$ covers the terminals in H. There is an edge between the terminals t_1 and t_2 in $C[H]$, even if they are composed nodes (cf. figure A4).

References

1. Hwang, F.; Richards, D.; Winter, P. *The Steiner tree problem*; Annals of discrete mathematics ; 53, North-Holland: Amsterdam, 1992.
2. Kruskal, J.B. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society* **1956**, *7*, 48–50.
3. Prim, R.C. Shortest connection networks and some generalizations. *The Bell System Technical Journal* **1957**, *36*, 1389–1401. https://doi.org/10.1002/j.1538-7305.1957.tb01515.x.
4. Takahashi, H.; Matsuyama, A. An approximate solution for the Steiner problem in graphs. *Math. Japonica* **1981**, *24*.
5. Kou, L.; Markowsky, G.; Berman, L. A fast algorithm for Steiner trees. *Acta Inform.* **1981**, *15*, 141–145. https://doi.org/10.1007/bf00288961.
6. Zelikovsky, A. An 11/6-Approximation Algorithm for the Steiner Problem on Graphs. In *Fourth Czechoslovakian Symposium on Combinatorics, Graphs and Complexity*; Neŝetril, J.; Fiedler, M., Eds.; Elsevier, 1992; Vol. 51, *Annals of Discrete Mathematics*, pp. 351–354. https://doi.org/10.1016/S0167-5060(08)70655-1.
7. Zelikovsky, A. Better approximation bounds for the network and Euclidean Steiner tree problems. Technical report, University of Virginia Charlottesville, VA, United States, 1996.
8. Karpinski, M.; Zelikovsky, A. New Approximation Algorithms for the Steiner Tree Problems. *Journal of Combinatorial Optimization* **1997**, *1*, 47–65. https://doi.org/10.1023/A:1009758919736.
9. Du, D.Z.; Zhang, Y.; Feng, Q. On better heuristic for Euclidean Steiner minimum trees. In Proceedings of the [1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science, 1991, pp. 431–439. https://doi.org/10.1109/SFCS.1991.185402.
10. Du, D.Z.; Zhang, Y. On better heuristics for Steiner minimum trees. *Mathematical Programming* **1992**, *57*, 193–202. https://doi.org/10.1007/BF01581080.
11. Berman, P.; Ramaiyer, V. Improved Approximations for the Steiner Tree Problem. *Journal of Algorithms* **1994**, *17*, 381–408. https://doi.org/10.1006/jagm.1994.1041.
12. Robins, G.; Zelikovsky, A. Tighter Bounds for Graph Steiner Tree Approximation. *SIAM Journal on Discrete Mathematics* **2005**, *19*, 122–134. https://doi.org/10.1137/S0895480101393155.
13. Byrka, J.; Grandoni, F.; Rothvoß, T.; Sanità, L. An improved LP-based approximation for steiner tree. In Proceedings of the Proceedings of the Forty-Second ACM Symposium on Theory of Computing, New York, NY, USA, 2010; STOC '10, p. 583–592. https://doi.org/10.1145/1806689.1806769.
14. Ljubić, I. Solving Steiner trees: Recent advances, challenges, and perspectives. *Networks* **2021**, *77*, 177–204, [https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.22005]. https://doi.org/10.1002/net.22005.
15. Warme, D. A new exact algorithm for rectilinear Steiner trees. In Proceedings of the Network Design: Connectivity and Facilities Location, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1997, Vol. 40, p. 357–395.

 doi:10.20944/preprints202602.1350.v1

16. Warme, D.M. Spanning trees in hypergraphs with applications to Steiner trees. PhD thesis, USA, 1998. https://doi.org/10.18130/V3ZG4B.
17. Prömel, H.J.; Steger, A. A New Approximation Algorithm for the Steiner Tree Problem with Performance Ratio 5/3. *Journal of Algorithms* **2000**, *36*, 89–101. https://doi.org/10.1006/jagm.2000.1086.
18. Kaufmann, M.; van Kreveld, M.; Speckmann, B. Subdivision Drawings of Hypergraphs. In Proceedings of the Graph Drawing; Tollis, I.G.; Patrignani, M., Eds., Berlin, Heidelberg, 2009; pp. 396–407.
19. Brazil, M.; Zachariasen, M., Steiner Trees in Graphs and Hypergraphs. In *Optimal Interconnection Trees in the Plane: Theory, Algorithms and Applications*; Springer International Publishing: Cham, 2015; pp. 301–317. https://doi.org/10.1007/978-3-319-13915-9_5.
20. Polzin, T.; Daneshmand, S.V. On Steiner trees and minimum spanning trees in hypergraphs. *Operations Research Letters* **2003**, *31*, 12–20. https://doi.org/10.1016/S0167-6377(02)00185-2.
21. Molnar, M. Construction d'arbres de diffusion multicast basée sur une modification de l'heuristique de Kruskal. Technical Report 3587, INRIA, France, 1998.
22. Marie, R.; Molnar, M. Arbre couvrant partiel pseudo-optimal pour diffusion multipoint. Technical Report 3636, INRIA, France, 1999.
23. Molnar, M. Hypergraphs to Cover Trees. *(submitted)* **2026**.