On Shortest Networks for Classes of Points in the Plane *†

Edmund Ihler Gabriele Reich Peter Widmayer ‡

Abstract

We are given a set P of points in the plane, together with a partition of P into *classes* of points; i.e., each point of P belongs to exactly one class. For a given network optimization problem, such as finding a minimum spanning tree or finding a minimum diameter spanning tree, we study the problem of choosing a subset P' of P that contains at least one point of each class and solving the network optimization problem for P', such that the solution is optimal among all possible choices for P'. We show that solving the minimum spanning tree problem for classes of points in the plane is NP-complete, where the distance between points is defined by any of the Minkowski metrics L_p, $1 \leq p \leq \infty$. In contrast, a class solution for the minimum diameter spanning tree problem can be computed in time $O(|P|^3)$.

By proving the NP-completeness of the minimum spanning tree class problem we also get some results for distance graphs. Here, computing a class solution for the minimum spanning tree problem is NP-complete, even under several restrictions, e.g., if the graph is part of a unit grid and is a tree, where the vertex degree and the number of vertices per class are both bounded by three. This is true even if the graph is a minimum spanning tree for its set of vertices.

*This work was partly supported by the Deutsche Forschungsgemeinschaft under grant Wid 810/5.

†Appeared in: *Computational Geometry - Methods, Algorithms and Applications, CG91*, No. 553 of *Lecture Notes in Computer Science*, pages 103–111. Springer, 1991.

‡Authors' address: Institut für Informatik, Universität Freiburg, Rheinstraße 10-12, W-79104 Freiburg, Germany; e-mail:ihler@informatik.uni-freiburg.de

1 Introduction

Consider the problem of connecting a number of existing local networks into a global network; for example, several local area computer networks might have to be connected into a metropolitan area or wide area network. Due to capacity, reliability and other technological considerations, we want the new links that need to be added to form a connected network of their own; most backbone networks in university campusses or companies are of this type. It is clearly sufficient for the new network to reach one of the computers in each local network. In this paper, we look at the problem of minimizing the total length of the new network (no Steiner points allowed), and at the problem of minimizing the length of the connection between any two local networks. In the plane, the length of a direct link is the distance between both points; we extend our considerations to other cost functions and the problem of identifying minimum subgraphs for given graphs.

In this paper, we consider network optimization problems for sets of points as well as for graphs. For our purposes, a set of n points $P = p_1, \dots, p_n$ in the plane is identified with its undirected, complete *Euclidean graph* $G = (V, E)$, where $V = P$, i.e., each point is a vertex in the graph, $E = V \times V$, and the length of edge (p_i, p_j) is the Euclidean distance between p_i and p_j. In fact, our results hold for any of the Minkowski distance functions instead of the Euclidean distance. In general, we are given a vertex set V and a partition of V into *classes* $V_1, \dots, V_c$, i.e. $\bigcup_{i=1}^{c} V_i = V$, $V_i \cap V_j = \emptyset$ for $i \neq j$, and a connected, undirected distance graph $G = (V, E)$. The objective is to compute a tree that contains at least one vertex of each class, a *class tree*. We consider two types of class trees: those that have minimum total length, the *minimum class trees MCT*, and those where the diameter is minimum, *minimum diameter class trees MDCT*. As usual, the diameter of a graph G is defined as the longest of the minimum paths among all pairs of vertices of G. If each class consists of a single vertex, then the problems described above reduce to the well-known problems of computing a minimum spanning tree and a minimum diameter spanning tree. Both problems can be solved in polynomial time for points in the plane, see [2, 5].

In sharp contrast to the minimum spanning tree problem, the problem of computing a minimum class tree is NP-complete. In Section 2, we will show that this holds for points in the plane as well as for arbitrary distance graphs. For specific types of distance graphs, problems that are NP-hard in general become solvable in polynomial — even linear — time, but minimum class tree remains NP-complete. In this sense, the latter is harder than Steiner's problem if the given graph is a tree with edge lengths restricted to 1, or if the tree is part of a unit grid. In each case the vertex degree and the number of vertices per class both can be restricted to at most three. The same result holds even if the input graph is not an arbitrary tree, but a minimum spanning tree for the given set of vertices.

In contrast to the NP-completeness of computing a minimum class tree, in Section 3 we will show that a minimum diameter class tree in the plane can be computed in polynomial time. To this end, we will present an algorithm that computes a minimum diameter class tree for a set of n points in the plane in time $O(n^3)$.

2 Minimum class tree is hard

In this section, we prove the NP-completeness of computing a minimum class tree for a finite set of points in the plane and for a connected, undirected distance graph. The former follows directly from the latter.

2.1 The minimum class tree of a graph

Let us now prove that the problem of computing a minimum class tree for a graph is NP-hard, even if the given graph is a tree.

Problem: MINIMUM CLASS TREE

Instance: A class graph $G = (V, E)$ with positive integer c, sets $V_1, \ldots, V_c$, such that $V_i \cap V_j = \emptyset$ for $i \neq j$, $1 \leq i, j \leq c$, and $\bigcup_{i=1}^{c} V_i = V$, and a positive integer k.

Question: Is there a class tree CT of G whose total length does not exceed k?

We will show the NP-completeness (in the strong sense) of MINIMUM CLASS TREE by reducing 3-satisfiability (3SAT) to it, where 3SAT is defined as follows (see [1, p. 46]):

Problem: 3SAT

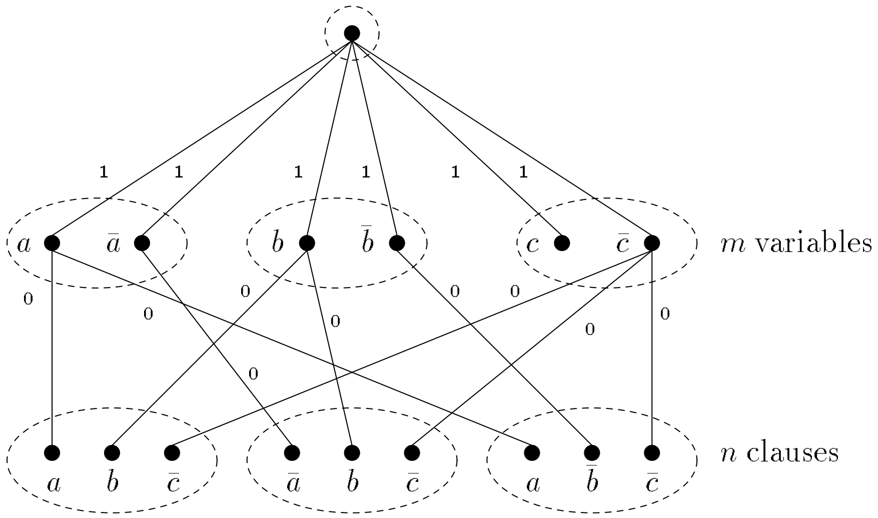
Instance: Collection $C = \{c_1, c_2, \ldots, c_n\}$ of clauses on a finite set $U = \{u_1, u_2, \ldots, u_m\}$ of variables such that $|c_i| = 3$ for $1 \leq i \leq n$.

Question: Is there a truth assignment for U that satisfies all the clauses in C?

Theorem 2.1: MINIMUM CLASS TREE is NP-complete, even if the given class graph G is a tree and each class contains at most three vertices.

Proof: Clearly, MINIMUM CLASS TREE is in NP, since a guess for CT can be verified in linear time to be a tree and a subgraph of G, contain one vertex of each class and have total length no more than k.

To see that MINIMUM CLASS TREE is NP-hard, even if G is a tree, consider the reduction of 3SAT to MINIMUM CLASS TREE as shown in Figure 1.



$$\ldots \wedge (a \vee b \vee \bar{c}) \wedge (\bar{a} \vee b \vee \bar{c}) \wedge (a \vee \bar{b} \vee \bar{c}) \wedge \ldots$$

Figure 1: Reduction of 3SAT to MINIMUM CLASS TREE

For each variable u_i, $1 \leq i \leq m$, there is a class of two vertices v_{i1} and v_{i2}, representing the assignment of *true* and *false* to u_i, respectively. For each clause c_i, $1 \leq i \leq n$, there is a class of three vertices v_{i1}, v_{i2}, v_{i3}, $m+1 \leq i \leq m+n$, representing the three literals in the clause. For each literal, the vertex is connected via an edge of length 0 to the vertex for the variable assignment that makes the literal true. All variable assignment vertices are connected via edges of length 1 to a single additional vertex v_c. Hence, there are $m+n+1 = c$ classes in the tree and the length of a minimum class tree is m.

Clearly, the graph defined by this reduction is a tree and each class contains at most three vertices; its size and the time for its computation are both polynomials in m and n. We show that the answer to the 3SAT question is the same as the answer to the MINIMUM CLASS TREE question for $k = m$.

Claim: The answer to MINIMUM CLASS TREE is "yes" iff the answer to 3SAT is "yes".

if: If the answer to 3SAT is "yes", there exists an assignment of values *true, false* to the variables u_i, $1 \leq i \leq m$, such that each clause is satisfied. For each variable u_i, $1 \leq i \leq m$, that is true, select the vertex v_{i1} to belong to the class tree CT, for each false variable u_j, $1 \leq j \leq m$, select v_{j2}. Enlarge CT by all edges incident to vertices already in CT and by the endpoints of these edges. This yields a tree containing one vertex of each of the classes representing variables (each variable is either true or false), the single vertex v_c, and — since at least one literal of each clause is true — at least one vertex of each of the classes representing clauses. Clearly, the length of this class tree is m. Hence, the answer to MINIMUM CLASS TREE is "yes".

only if: If the answer to MINIMUM CLASS TREE is "yes", there exists a tree CT, with total length at most k, containing at least one vertex of each class. Obviously, v_c must be in CT. Because of the total length of CT and the length 1 of the edges between v_c and the variable assignment vertices v_{ij}, $1 \leq i \leq m$, $1 \leq j \leq 2$, at most m of the $2m$ vertices v_{ij} may be in CT. Moreover, each of the classes V_i must be represented by a vertex in CT; therefore, exactly one variable assignment vertex of each of the classes V_i, $1 \leq i \leq m$, is in CT. At least one vertex of each class V_i, $m+1 \leq i \leq m+n$, needs to be in T. Any such vertex v in CT implies that the variable assignment vertex v' where (v, v') is an edge of G is also in CT. Now assign u_i the value *true* if v_{i1} is in CT, and *false* if v_{i2} is in CT, $1 \leq i \leq m$. Hence, for each clause at least one literal is true, and so the answer to 3SAT is "yes".

This proves the claim and the theorem. □

From the proof it is clear that G is a tree and that each class contains no more than three vertices. Moreover, a straightforward modification of the proof allows to restrict the degree of each vertex to three.

2.2 The minimum class tree in the plane

Recall that the *Euclidean graph* $G = (P, E)$ is the complete graph that is induced by a set P of points $p_1, \ldots, p_n$ in the plane, where the length of an edge (p_i, p_j) is the Euclidean distance $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ between p_i and p_j. A minimum class tree for a Euclidean graph is called a *geometrical minimum class tree gMCT.*

We will show that computing a minimum class tree for a Euclidean graph is NP-hard.

Problem: GEOMETRICAL MINIMUM CLASS TREE

Instance: A set P of points in the plane, a positive integer c, a partition of P into subsets $P_1, \ldots, P_c$, such that $P_i \cap P_j = \emptyset$ for $i \neq j$, $1 \leq i, j \leq c$, and $\bigcup_{i=1}^{c} P_i = P$, a positive integer k.

Question: Is there a class tree CT of the Euclidean graph $G = (P, E)$ with total length at most k?

We will show the NP-completeness of GEOMETRICAL MINIMUM CLASS TREE by a reduction of 3SAT to GEOMETRICAL MINIMUM CLASS TREE, similar to the one presented in Section 2.1. Note that NP-hardness for GEOMETRICAL MINIMUM CLASS TREE does not follow directly from NP-hardness for MINIMUM CLASS TREE, since the former is quite a special case of the latter.

Theorem 2.2: GEOMETRICAL MINIMUM CLASS TREE is NP-complete, even if the number of points per class is restricted to at most three.

Proof: By the same arguments as for MINIMUM CLASS TREE, GEOMETRICAL MINIMUM CLASS TREE is in NP. It remains to be shown that GEOMETRICAL MINIMUM CLASS TREE is NP-hard.

Consider the reduction of 3SAT from Section 2.1 and the NP-complete variant of 3SAT, where for each $u \in U$ there are most five clauses that contain either u or $\bar{u}$ [1, p. 259].

For each variable u_i create two lists l_{i1} and l_{i2} of ten vertices each, l_{i1} for the assignment of *true* to u_i, l_{i2} for u_i *false*. Connect with every other vertex of l_{i1} a vertex representing an input literal u_i, with every other vertex of l_{i2} a vertex representing an input literal $\bar{u}_i$. Since for each variable there are at most five input literals, the ten vertices of each of l_{i1} and l_{i2} will suffice for this purpose. As in the proof of Theorem 2.1, let the vertices representing literals of the same clause form a class. Also, the j-th vertices of lists l_{i1} and l_{i2}, $1 \leq i \leq m$, $1 \leq j \leq 10$, form a class. Now create a list l_c of $6m - 2$ vertices, each belonging to a new class. Connect l_{11} with the first vertex of l_c, and then connect all remaining lists l_{ij}, $1 \leq i \leq m$, $1 \leq j \leq 2$, with every third vertex of l_c. Embed the tree constructed in this way into a unit grid as it is shown in Figure 2.

This tree consists of $n + 10m + 6m - 2$ classes; now, let $k = n + 10m + 6m - 2 - 1$.

Claim: The answer to GEOMETRICAL MINIMUM CLASS TREE is "yes" iff the answer to 3SAT is "yes".

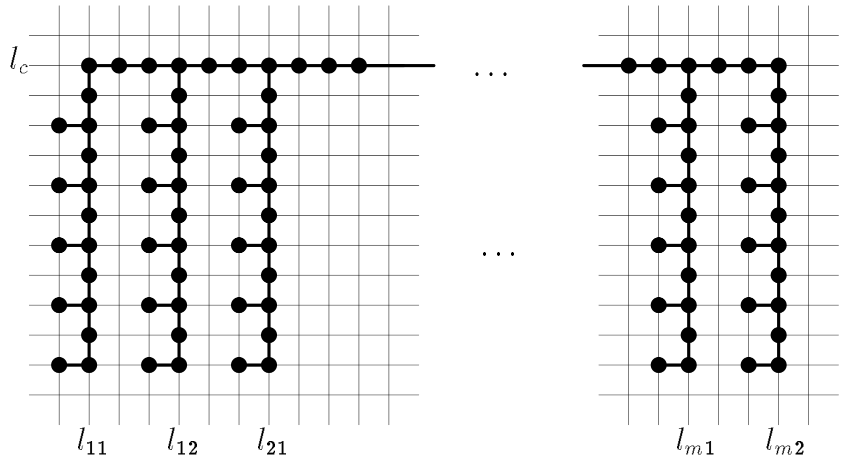


Figure 2: Reduction of 3SAT to Geometrical Minimum Class Tree

if: Our arguments are similar to those in the proof of Theorem 2.1. There must exist an assignment of values *true, false* to the variables u_i, $1 \leq i \leq m$, satisfying each clause, since the answer to 3SAT is "yes". For each variable u_i that is true, select the corresponding list l_{i1} to belong to CT, for each false variable u_j select l_{j2}, $1 \leq i, j \leq m$. Connect these lists with the complete list l_c. Eventually, for each clause c_i connect the vertex representing the first true literal of c_i with CT by an edge of length 1. Since each clause is satisfied, exactly one vertex of each of the n classes representing clauses belongs to CT. Moreover, each of the $6m - 2$ classes of l_c is in CT and one vertex of each of the $10m$ classes for the m variables is in CT, too, since either l_{i1} or l_{i2}, $1 \leq i \leq m$, completely belongs to CT. Since all edge lengths in CT are 1, the total length of CT is $n + 10m + 6m - 2 - 1$, so the answer to Geometrical Minimum Class Tree is "yes".

only if: The answer to Geometrical Minimum Class Tree is "yes", hence, a class tree CT with total length at most $n + 10m + 6m - 2 - 1$ exists. Since all edge lengths are at least 1, the number of vertices in CT is at most $n + 10m + 6m - 2$. Because the number of classes also is $n + 10m + 6m - 2$, it follows that for each class exactly one vertex belongs to CT and that each edge in CT has length 1. Therefore, for any i, $1 \leq i \leq m$, either the complete list l_{i1} or l_{i2} may be in CT, but not parts of both or even both completely. Moreover, for each of the n classes representing clauses, the single vertex v that belongs to CT must be connected with one of the lists l_{ij}, $1 \leq i \leq m$, $1 \leq j \leq 2$, that belongs to the tree, otherwise the length of the edge connecting v with CT is at least 2. Clearly, l_c must belong to CT completely. It follows that we can assign the variable u_i the value *true*, if l_{i1} belongs to CT, and *false* otherwise, $1 \le i \le m$, thereby satisfying each of the n clauses. Hence, the answer to 3SAT is "yes". □

Corollary 2.1: GEOMETRICAL MINIMUM CLASS TREE is NP-complete if the distance between points in the plane is defined by any of the Minkowski metrics L_p, $1 \le p \le \infty$.

Proof: From Figure 2 it follows that for any metric L_p, $1 \le p \le \infty$, the distance defined by L_p between any two vertices in the graph is at least 1. Since only edges of length 1 belong to the minimum class tree constructed in the proof of Theorem 2.2, the corollary follows. □

With the proof of Theorem 2.2 also the following corollary is self-evident.

Corollary 2.2: MINIMUM CLASS TREE is NP-complete for class graphs that are part of a unit grid and where both, the vertex degree and the number of vertices per class, is restricted to at most three.

Furthermore, it follows immediately that MINIMUM CLASS TREE is NP-complete also for trees with edge lengths restricted to 1.

2.3 The minimum class tree of a minimum spanning tree

Since computing a minimum class tree for a graph as well as for points in the plane is NP-complete, approximate rather than exact solutions are of interest [3, 4, 6]. A straightforward approach to approximate a minimum class tree is first to compute a minimum spanning tree for the vertices of the given graph or for the given set of points in the plane, and then to compute a minimum class tree MCT for the given minimum spanning tree MST. NP-completeness of MCT for an input tree does not immediately imply NP-completeness of MCT for an input MST. The following corollary, however, shows that MCT is a hard problem, even for this specific input.

Corollary 2.3: The decision whether there exists a minimum class tree of length at most k is NP-complete for the following input graphs:

1. class graphs that are minimum spanning trees for the vertex sets of complete graphs;
2. class graphs that are minimum spanning trees for arbitrary sets of points in the plane, where the distance between points is defined by any of the Minkowski metrics L_p, $1 \le p \le \infty$;

3. class graphs that are minimum spanning trees of graphs that are part of a unit grid.

Proof:

ad 1) Consider the proof of Theorem 2.1 and the graph G shown in Figure 1. Let G' be the complete graph of G, obtained by adding to G all missing edges and assigning to each of the new edges a length greater than the sum of all edge lengths of G. Then, G is a minimum spanning tree for the given set of vertices.

ad 2), 3) Consider the proof of Theorem 2.2 and the graph G shown in Figure 2. The number of vertices in G is $3n+20m+6m-2$. Since the minimum distance between any two vertices is 1, a minimum spanning tree for G must have length $3n+20m+6m-2-1$. But this is exactly the total length of G; hence, G is a minimum spanning tree for the given set of vertices. Obviously, this result holds for sets of points in the plane with any of the Minkowski metrics L_p, $1 \leq p \leq \infty$, as well as for graphs that originate from a unit grid. □

3 Minimum diameter class tree is easy

As mentioned in Section 1, when connecting networks by a tree it sometimes will be more appropriate to minimize the length of connections between pairs of networks rather than minimizing the total length of the connecting network. Hence, the connecting network should be a minimum diameter class tree.

In this section, we will describe how to find a class tree with minimum diameter for n points and c classes in the plane, a *geometrical minimum diameter class tree gMDCT*, in time $O(n^3)$.

First, consider the problem of computing the *geometrical minimum diameter spanning tree gMDST* for a Euclidean graph without classes. In [2], it was shown that the diameter of a *gMDST* with the minimum number of edges contains at most three edges and can be computed in time $O(n^3)$ for n points in the plane.

Obviously, for class graphs also, the diameter of a *gMDCT* with the minimum number of edges contains at most three edges. Hence, the *gMDCT* has as its center either a single point, called a *monopole*, or two points, called a *dipole*. A *gMDCT* then can be determined by computing for each possible monopole and for each possible dipole a class tree with minimum diameter and selecting the best of these.

To compute a monopolar tree, first the distance of each point to the monopole is computed. In a second step, for each class the point closest to the monopole is determined. Let p be the monopole and let p' and p'' be the two points of different classes that have the largest distances to the monopole among the points determined in the second step. Then, the diameter is the sum of the distances between p and p' and between p and p''. Computing one monopolar tree clearly takes time $O(n)$.

The computation of a dipolar tree is slightly more complicated. Let the dipole be two points p and q. We must compute two circles, with centers at p and q, respectively, such that both circles together contain a point of each class and the sum of the radiusses is minimal. Then, the diameter is the sum of the radiusses of the two circles plus the length of the edge between p and q. The two circles can be found as follows: Let the points be given in two lists, one sorted by distance to p, the other sorted by distance to q. Eliminate for each class all but one point of smallest distance from the p-list and from the q-list; the resulting lists each contain c points. Now, scan the p-list in increasing order. For each point p' in the list, a class is said to be contained in the circle of p' around p if a point of that class is in that circle. Now, determine the circle with smallest radius around q that contains a point for each class not contained in the circle of p' around p. For the sequence of all these points p', the corresponding circle around q can be determined by scanning the q-list once in decreasing order, since the circle around p grows monotonically for increasing distance of p' to p, and the circle around q shrinks monotonically. During the scan, for a point q' in the q-list we check whether the class of q' is already contained in the circle of p' around p. By marking each class contained in the circle of p' around p, each check can be performed in constant time. Hence, the scan of the lists costs only $O(n)$ time for each dipole. If the class of q' is already contained in the circle around p, q' is skipped and the next point in the q-list is inspected. For the smallest circle around q containing points of all classes that are not contained in the circle of p' around p, the diameter is computed. That is, for each point in the p-list a dipolar class tree and its diameter are computed. For the dipole p, q, we keep track of the smallest of these trees.

Since there are n points that can be the center of a monopolar tree and n^2 pairs of points that can be the centers of a dipolar tree, it takes time $O(n^3)$ to compute the set of class trees in which a class tree with minimum diameter and minimum number of edges on the diameter must be contained.

We get the following theorem.

Theorem 3.1: A geometrical minimum diameter class tree for a Euclidean graph with n points can be computed in time $O(n^3)$.

4 Conclusion

We have considered the problem of computing a global network that connects local networks. The new network should either have minimum total length or minimum diameter. We modelled the problem by a more general type of problem: a given set of points in the plane or of vertices of a graph is partitioned into classes. Each class represents a property common to all points in a class. Finding a global network of minimum total length then equals computing a minimum class tree, i.e., a tree of minimum total length that contains at least one point of each class.

We showed that computing a minimum class tree is NP-hard for points in the plane, where the distance between points is defined by any of the Minkowski metrics L_p, $1 \leq p \leq \infty$. The same result holds for distance graphs, even under several restrictions, like e.g., if the given graph is a tree, where the vertex degree is at most three and where each class contains no more than three vertices.

Finding a network with minimum diameter that connects all local networks can be modelled by computing a tree with minimum diameter that contains at least one point of each of the classes. We showed that a class tree with minimum diameter can be computed in time $O(n^3)$ for n points in the plane.

There are other application areas where problems can be modelled by classes of vertices of a graph or classes of points in the plane. In the wire routing phase in physical VLSI design, for instance, components on a chip may be flipped or rotated without changing the placement of the components on the chip, generating a class of possible pin positions (see e.g. [6]).

References

[1] M.R. Garey, D.S. Johnson:
Computers and intractability: a guide to the theory of NP-completeness, Freeman, New York, 1979.

[2] J.-m. Ho, D.T. Lee, C.-H. Chang, C.K. Wong:
Bounded-diameter minimum spanning trees and related problems, Symp. on Comp. Geometry, 1989, 276–282.

[3] E. Ihler:
Bounds on the quality of approximate solutions to the Group Steiner Problem, Graph-Theoretic concepts in Computer Science, WG90, Lecture Notes in Computer Science, Springer, 1991.

[4] E. Ihler:
The complexity of approximating the class Steiner tree problem, Tech. Report, Institut für Informatik, Universität Freiburg, Germany, 1991.

[5] F.P. Preparata, M.I. Shamos:
Computational Geometry: an introduction, Springer, New York, 1985.

[6] G. Reich, P. Widmayer:
Beyond Steiner's problem: a VLSI oriented generalization, Graph-Theoretic concepts in Computer Science, WG89, Lecture Notes in Computer Science, Vol. 411, Springer, 1990, 196–210.