Discrete Mathematics and Theoretical Computer Science **3**, 1998, 1–10

$\mathcal{NP}$–Completeness Results for Minimum Planar Spanners[†]

Ulrik Brandes and Dagmar Handke[‡]

University of Konstanz, Faculty of Mathematics and Computer Science, D-78457 Konstanz, Germany.
email: {*Ulrik.Brandes,Dagmar.Handke*}*@uni-konstanz.de*

received 10 Sep 1997, *revised 13 Mar 1998*, *accepted 3 Jun 1998*.

For any fixed parameter $t \geq 1$, a *t–spanner* of a graph G is a spanning subgraph in which the distance between every pair of vertices is at most t times their distance in G. A *minimum* t–spanner is a t–spanner with minimum total edge weight or, in unweighted graphs, minimum number of edges. General t–spanners and their variants have multiple applications in the field of communication networks, distributed systems, and network design. In this paper, we prove the $\mathcal{NP}$–hardness of finding minimum t–spanners for planar weighted graphs and digraphs if $t \geq 3$, and for planar unweighted graphs and digraphs if $t \geq 5$. We thus extend results on that problem to the interesting case where the instances are known to be planar. We also introduce the related problem of finding minimum *planar* t–spanners and conclude its $\mathcal{NP}$–hardness for similar fixed values of t.

Keywords: graph spanners, planar graphs, $\mathcal{NP}$–completeness

1 Introduction

A *t–spanner* of a graph G is a spanning subgraph S in which the distance between every pair of vertices is at most t times their distance in G. The main idea of this concept is to find a subgraph of a given graph G that is sparse, but still guarantees a so–called *stretch factor* on the vertex–to–vertex distances of G. The stretch factor will be bounded by a constant independent of the size of G. Observe that the minimum spanning tree does not necessarily meet this specification.

The concept of spanners has been introduced by Peleg and Ullman in [12], where they used spanners to synchronize asynchronous networks. One of many other applications for spanners are communication networks, where one is interested in finding a sparse subnetwork that nevertheless guarantees constant delay factors. A survey of some results on the existence and efficient constructibility of (sparse) spanners is given in [11]. Further results and discussions concerning t–spanners and variants thereof can be found in [13].

In most applications, the sparseness of a spanner is crucial. The problem of finding t–spanners with a minimum number of edges has been shown to be $\mathcal{NP}$–hard for most values of t by Cai in [2]. Therefore,

[†]Research partially supported by DFG under grant Wa 654/10-1.
[‡]corresponding author

subsequent efforts have concentrated on finding spanners that are maybe not minimum, but sufficiently sparse (see for example [1]). Very recently, [14] have proven the $\mathcal{NP}$–hardness of the problem even for restricted, unweighted graph classes such as chordal, split, bipartite, or degree-constraint graphs. Several authors consider variants of t–spanners. In [3], Cai and Corneil deal with *tree t–spanners* (i.e. t–spanners that are trees) and also examine the complexity status of the corresponding decision problem. Liestman and Shermer introduce the notion of *additive* spanners, which employ an additive instead of multiplicative stretch function on the distances [8].

Here we consider general spanners in *planar* graphs (either weighted or unweighted, directed or undirected), i.e. we restrict the set of input instances. We thereby (partially) settle a question raised in [2]. We also introduce the notion of *planar t–spanners*. These are subgraphs, which are not only t–spanners, but also planar, no matter whether the original graph is planar or not.

This paper is organized as follows: After introducing some basic notation and the examined problems, our results of $\mathcal{NP}$–completeness are stated in Sect. 2. Proofs of these in unweighted, weighted, and directed graphs make up for Sects. 3, 4, and 5, respectively.

2 Problems and Results

In what follows $G = (V, E; w)$ (respectively, $G = (V, A; w)$) denotes a simple, weighted undirected (directed) graph with vertex set V, edge set E (arc set A), and edge weights $w : E \to \mathbb{R}^+$ ($w : A \to \mathbb{R}^+$). If all edges have unit weight, i.e. all weights are equal to 1, the graph is said to be *unweighted*. A directed graph (*digraph*) is said to be an *oriented* graph, if it does not contain a cycle of two arcs. For simplicity, we will use the terminology for undirected graphs throughout most of this paper. The terms are naturally extended to digraphs. Since spanners of each connected component can be determined independently, we only consider connected graphs. The *length* of a path is the sum of the weights of its edges. The *distance* between two vertices u and v in G, i.e. the length of the shortest (directed) path, is denoted by $d_G(u, v)$.

2.1 Minimum t–Spanners in Planar Graphs

Definition 1 (t–spanner) *For any parameter $t \geq 1$, a spanning subgraph $S = (V, E'; w)$ with $E' \subseteq E$ is a t–*spanner *of an edge-weighted graph $G = (V, E; w)$, if $d_S(u, v) \leq t \cdot d_G(u, v)$ for all $u, v \in V$.*

The parameter t is called *stretch factor*. We say that an edge $e \in E$ is *covered* (by an edge $f \in S$), if in S there exists a path of length at most $t \cdot w(e)$ (and containing f) that connects the endpoints of e. Note that in unweighted graphs every t–spanner is also a $\lfloor t \rfloor$–spanner, while there is no such correspondence in weighted graphs, even if all edges have integer weights. In order to prove that a given spanning subgraph is a t–spanner, we do not have to consider all pairwise distances of the vertices. It is sufficient to only look at edges of the original graph that are not part of the spanning subgraph (see [3]).

A t–spanner is called a *minimum* t–spanner of a weighted graph G, if it has minimum total edge weight among all t–spanners of G. The corresponding decision problem is defined as follows:

Minimum t–Spanner Problem ($\mathbf{MinS_t}$)
Given: A graph G with associated (positive) edge weights and a positive value W.
Problem: Does G contain a t–spanner with total edge weight at most W?

Obviously, for an unweighted graph, the only 1–spanner is the graph itself. For a weighted graph, Hakimi and Yau [4] proved that there is a unique 1–spanner with a minimal number of edges. From

[3], we know that this must also be the unique minimum 1–spanner, and that it can be determined in polynomial time. The $\mathcal{NP}$–completeness of MinS_t for the remaining values of t has been established in [2] and [3]. Here we will show that the problem remains $\mathcal{NP}$–complete for most values of t when G is restricted to be planar. In particular, we prove the following theorem.

Theorem 2

1. *For any fixed integer $t \geq 5$, $MinS_t$ is $\mathcal{NP}$–complete for undirected, unweighted, planar, biconnected graphs.*
2. *For any fixed integer $t \geq 3$, $MinS_t$ is $\mathcal{NP}$–complete for undirected, weighted, planar, biconnected graphs with edge weights equal to 1 or 2.*
3. *For any fixed integer $t \geq 5$ ($t \geq 3$), $MinS_t$ is $\mathcal{NP}$–complete for unweighted (weighted) planar oriented graphs.*

The proofs of the three parts of the theorem are given in the next sections. All three of them are transformations from the Planar Satisfiability Problem with three literals in each clause, and they can be viewed as modifications of each other. At the end of Sect. 4 it will be easy to see how our construction can be adjusted to allow arbitrary rational values of $t \geq 3$ in the weighted case. Thus Theorem 2 also holds for rational stretch factors.

2.2 Minimum Planar t–Spanners

Instead of restricting the input graph we now consider restrictions on the spanning subgraph by introducing planar t–spanners:

Definition 3 (planar t–spanner) *For any parameter $t \geq 1$, a spanning subgraph $S = (V, E'; w)$ with $E' \subseteq E$ is a* planar t–spanner *of a weighted graph $G = (V, E; w)$, if $d_S(u, v) \leq t \cdot d_G(u, v)$ for all $u, v \in V$, and S is planar.*

The corresponding *Minimum Planar* t*–Spanner Problem* is denoted by MinPS_t. First, consider the *unweighted* case: Using a linear time planarity test, it is clear that MinPS_1 is in $\mathcal{P}$ for unweighted graphs. On the other hand, it is $\mathcal{NP}$–complete to decide whether an unweighted graph contains a tree t–spanner, i.e. a t–spanner which is a tree, if $t \geq 4$ [3]. Observe that spanning trees are planar spanning subgraphs with the least possible number of edges.

For weighted graphs the situation is different: As mentioned above, the unique 1–spanner with a minimal number of edges also is the unique minimum 1–spanner, and can be determined in polynomial time. Since all edge weights are positive, and every subgraph of a planar graph is planar, a minimum planar 1–spanner has a minimal number of edges. Therefore a minimum planar 1–spanner has to be identical to the minimum 1–spanner, and we can conclude that MinPS_1 is in $\mathcal{P}$ for weighted graphs by testing the minimum 1–spanner for planarity.

In [3], the $\mathcal{NP}$–completeness of the Tree t–Spanner Problem for $t > 1$ in weighted graphs is proven. By a close look at the transformation used there and by an appropriate choice of the bound on the total weight of a planar t–spanner, the proof can be modified to show the $\mathcal{NP}$–completeness of MinPS_t for $t > 1$ in weighted, undirected graphs. Altogether, we get the following corollary:

Corollary 4

1. *For any fixed rational number $t \geq 4$, $MinPS_t$ is $\mathcal{NP}$–complete for unweighted graphs.*
2. *For any fixed rational number $t > 1$, $MinPS_t$ is $\mathcal{NP}$–complete for weighted graphs.*

Observe that MinS_t and MinPS_t are the same for planar instances.

t	**MinS$_t$**, general graphs [2, 3]		**MinPS$_t$**, general graphs		**Min(P)S$_t$**, planar graphs	
	unweighted	weighted	unweighted	weighted	unweighted	weighted
1	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$
(1,2)		$\mathcal{NPC}$		$\mathcal{NPC}$		?
[2,3)	$\mathcal{NPC}$	$\mathcal{NPC}$	?	$\mathcal{NPC}$	?	?
[3,4)	$\mathcal{NPC}$	$\mathcal{NPC}$	?	$\mathcal{NPC}$	?	$\mathcal{NPC}$
[4,5)	$\mathcal{NPC}$	$\mathcal{NPC}$	$\mathcal{NPC}$	$\mathcal{NPC}$	?	$\mathcal{NPC}$
$[5,\infty)$	$\mathcal{NPC}$	$\mathcal{NPC}$	$\mathcal{NPC}$	$\mathcal{NPC}$	$\mathcal{NPC}$	$\mathcal{NPC}$

Tab. 1: The complexity status of MinS$_t$ and MinPS$_t$ in undirected graphs

2.3 Summary of Results

Table 1 summarizes the results for the complexity status of the problems considered in this paper for undirected graphs in comparison to MinS$_t$ (as shown in [2] and [3]). The results are listed for both the weighted and the unweighted case. A question mark indicates that the complexity status is unknown.

Very recently, Kortsarz has shown in [5] that, for general undirected, unweighted graphs and every $t \geq 2$, the Minimum t–Spanner Problem is as hard to approximate as the Set Cover Problem. The approximability status of the problems considered here is still open. For one of the open cases, MinS$_2$ on planar graphs, the $|V|/|E|$–approximation algorithm of Kortsarz and Peleg [7] for general undirected, unweighted graphs results in a constant approximation algorithm for planar input graphs.

3 MinS$_t$ for Unweighted, Planar Graphs

In this section, we prove part 1 of Theorem 2, so all graphs are unweighted and planar. The other parts are proven along the same lines. Part of the proof modifies ideas of [2].

Let $t \geq 5$ be an arbitrary fixed integer. Clearly, MinS$_t$ is in $\mathcal{NP}$, since the test whether a spanning subgraph S is a t–spanner can be done in polynomial time. To show the $\mathcal{NP}$–completeness we transform the Planar 3–Satisfiability Problem to MinS$_t$:

Planar 3–Satisfiability Problem (P3SAT)

Given: A set U of variables, and a collection C of clauses over U with $|c| = 3$ for all $c \in C$. Furthermore the bipartite graph $G = (V, E)$ where $V = U \cup C$ and $E = \{\{x, c\} : x \text{ or } \overline{x} \text{ occurs in } c\}$ is planar.

Problem: Is there a satisfying truth assignment for C?

The $\mathcal{NP}$–completeness proof for this problem can be found in [9]. We use the planarity of the underlying graph of P3SAT to construct a planar graph in which we can easily determine the minimum t–spanner.

3.1 Forcing Edges into a Minimum t–Spanner

For the construction of the instance of MinS$_t$, we use the fact that we can force edges to be in every minimum t–spanner by adding some additional edges. This concept has appeared in [2] and will be used extensively.

Lemma 5 *Let e be an arbitrary edge of an unweighted graph G, and let G' be the graph constructed from G by adding two distinct paths P_1 and P_2 of length t (all internal vertices of P_1 and P_2 are new vertices) between the ends of e. Then for any minimum t–spanner S of G', edge e belongs to S.*

The two auxiliary paths P_1 and P_2 are called *forcing paths*, edge e is called *forced edge*. A *forced l–component* is a simple path of length l consisting of l forced edges together with their forcing paths. A minimum t–spanner of a forced l–component contains exactly $l \cdot (2 \cdot (t-1) + 1) = l(2t-1)$ edges: the l forced edges and $t-1$ edges from each forcing path.

3.2 Construction of the Instance

We start from the planar, embedded graph underlying the given instance (U, C) of P3SAT, and extend the variable and clause vertices to form *variable components* and *clause components*. These components are combined to form *truth assignment testing components* which reflect the relationship between the satisfiability of a clause and the existence of a minimum t–spanner. Finally, we compute a bound on the number of edges for the Minimum t–Spanner Problem.

Variable Components. The key idea behind the variable component is that each of its possible minimum t–spanners reflects exactly one truth assignment for its corresponding variable. For each variable $x \in U$, we construct a variable component T_x as follows. Let k be the number of (positive and negative) occurrences of the variable x in all clauses.

1. Create a central vertex x^*.
2. For each occurrence of x in a clause c, create in this order a block of four new vertices $x_1^{(c)}$, $\overline{x}_1^{(c)}$, $x_2^{(c)}$, and $\overline{x}_2^{(c)}$. The resulting $4k$ new vertices are called *literal vertices*. The blocks are arranged circularly around x^* according to the embedding of the underlying graph of the instance of P3SAT.
3. Connect each pair of neighboring literal vertices by a forced $(t-1)$–component such that a circle of $4k$ forced $(t-1)$–components is formed altogether.
4. Connect x^* with all literal vertices by an edge, called *literal edge*. An edge $\{x_i^{(c)}, x^*\}$ is called *positive* literal edge, an edge $\{\overline{x}_i^{(c)}, x^*\}$ is called *negative* literal edge.
5. Create $4k$ new auxiliary vertices, one between each pair of neighboring literal edges. Connect each of these by an *auxiliary* edge with x^* and by two distinct forced $(t-1)$–components with its neighboring literal vertices. Their literal edges are then called *associated* literal edges of the auxiliary edge and vice versa.

Figure 1 illustrates this construction. For readability, the symbolic representation in Figure 1(b) is used later on when larger portions of the graph are drawn. The following lemma shows that the literal edges in a minimum t–spanner are *consistent*. As a consequence, the number of edges of a minimum t–spanner of T_x is $4 \cdot 3k \cdot (t-1) \cdot (2t-1) + 2k$.

Lemma 6 *Any minimum t–spanner of a variable component T_x contains either all positive or all negative literal edges.*

Proof: Let S be an arbitrary minimum t–spanner of T_x. Then S contains all forced edges and $t-1$ edges from each forcing path. Observe that these edges together with either all $2k$ positive or all $2k$ negative

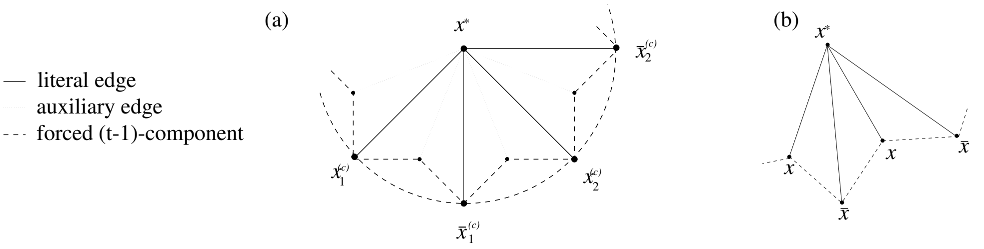

Fig. 1: (a) Part of the variable component T_x for the variable x occurring in clause c, (b) its symbolic representation

literal edges form a t–spanner. Thus S can contain at most $2k$ edges out of the $8k$ literal and auxiliary edges.

By construction of the variable component, both associated auxiliary edges and both neighboring negative (resp. positive) literal edges are covered by a positive (resp. negative) literal edge in S. But, by an auxiliary edge in S, only the associated literal edges are covered.

Now assume that S contains an auxiliary edge. Then S also contains either the next auxiliary edge, too, or the next not associated literal edge. In total, this leads to more than $2k$ additional edges and thus contradicts the minimality of S. Similarly, assume that S contains two inconsistent literal edges. Then there must be at least one auxiliary edge belonging to S or more than $2k$ literal edges to cover all other edges. Again, this contradicts the minimality of S. Thus S contains exactly every other literal edge. □

Clause Components. The clause component for each clause $c \in C$ is basically a quadrilateral consisting of four *clause vertices* 1, 2, 3, and 4, where the sides are formed by distinct forced $(t-2)$–components. Vertices 1 and 3 are connected by an additional edge, called the *clause edge*. See Fig. 2(a) for an example. Our construction is a bit more complex than actually needed in the unweighted case, but will not have to be changed much when being modified for the weighted and the directed case. Observe that any minimum t–spanner for $t \geq 5$ of such an isolated clause component must contain the clause edge.

Truth Assignment Testing Components. We combine the clause components with the variable components according to the given clauses by identifying vertices. Three sides of the quadrilateral in the clause component each correspond to a literal in the corresponding clause. The fourth side is used to

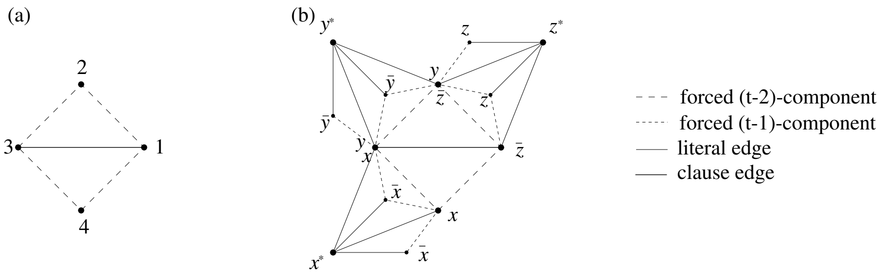

Fig. 2: (a)A clause component, and (b) the truth assignment testing component for clause $c = x \vee y \vee \overline{z}$ using the symbolic representation for relevant blocks of the variable components

make the arguments symmetrical. The endpoints of each such side of the quadrilateral are thus identified with the two corresponding literal vertices of the corresponding block in the variable component: if clause c contains the positive literal we use the positive literal vertices $x_i^{(c)}$, and $\overline{x}_i^{(c)}$ otherwise. See Fig. 2(b) for an example. Note that the combination of the variable components with the clause components does not affect the validity of Lemma 6.

Lemma 7 *For any fixed integer $t \geq 5$, a minimum t–spanner S of a truth assignment testing component contains the clause edge if and only if S contains no pair of consistent literal edges that is adjacent to the clause edge.*

Proof: If S does not contain a pair of literal edges that is adjacent to the clause edge then every path connecting the endpoints of the clause edge in S either uses the clause edge or has length at least $2(t-2) > t$, if $t \geq 5$.

For the other direction, assume that S contains a pair of adjacent consistent literal edges. Then this provides a shortcut for one of the forced $(t-2)$–components, and thus there is a path of length $2+(t-2) = t$ in S connecting the endpoints of the clause edge. Hence the clause edge is covered. □

Thus, the number of edges in a minimum t–spanner of such a truth assignment testing component reflects the truth value of the corresponding clause. This completes the construction of the graph. All isolated components are planar, and since we start from an instance of P3SAT, the whole graph is planar. It is also easily seen that the instance is biconnected, and can be constructed in polynomial time.

Choice of *W*. According to Lemmas 6 and 7, we set W, the bound on the number of edges in a t–spanner, to $W = 6m + 36m(2t-1)(t-1) + 4m(2t-1)(t-2)$, where m is the number of clauses of the instance of P3SAT.

3.3 Equivalence of the Problems

In this subsection, let (U, C) be an instance of P3SAT, and (G, W) the instance for MinS_t constructed as described above. We will show that there is a satisfying truth assignment for (U, C), if and only if G has a t–spanner with at most W edges.

Lemma 8 *If the set of clauses C of (U, C) is satisfiable, then there exists a planar t–spanner of G with at most W edges.*

Proof: Suppose that the set of clauses C is satisfiable, and let θ be a satisfying truth assignment. From this we construct the subgraph S of G as follows:

1. S contains all forced edges.
2. S contains $t-1$ arbitrarily chosen edges from each forcing path.
3. For each variable $x \in U$, S contains all positive literal edges if $\theta(x)$ is *true*, and all negative literal edges otherwise.

By this construction, S trivially is a spanning subgraph. S consists of forced edges from the variable components, edges from the forcing paths of the variable components, forced edges from the clause components, edges from the forcing paths of the clause components, and literal edges. Altogether S contains exactly $W' = 6m + 36m(2t-1)(t-1) + 4m(2t-1)(t-2) = W$ edges.

It remains to prove that S is a t–spanner of G: We have to show that for every edge not contained in S, there exists a path of length at most t connecting the endpoints of that edge. This is obvious for the variable components. For the clause edges observe that, since θ is a satisfying truth assignment, there is at least one literal in each clause that is true. Due to the construction of S we thus have at least one adjacent pair of literal edges in each clause component. From Lemma 7 it follows that S is a t–spanner. □

Using the next lemma, Lemma 10 completes the proof of part 1 of Theorem 2.

Lemma 9 *Any minimum t–spanner S of G contains at least W edges.*

Proof: Any t–spanner S of G must contain all forced edges and $t-1$ edges from each forcing path. By Lemma 6, S contains at least either all positive or all negative literal edges for each variable component. This sums up to W. □

Lemma 10 *If G has a t–spanner with at most W edges, then there exists a satisfying truth assignment for (U, C).*

Proof: Suppose S is a t–spanner of G with at most W edges. Then by Lemma 9, S is a minimum t–spanner and contains exactly W edges. All forced edges and the corresponding edges from the forcing paths must be in S. Hence, there remain only $6m$ further edges which can only be consistent literal edges (by Lemma 6). Thus, we can uniquely define a truth assignment θ by setting, for each $x \in U$, $\theta(x) = $ *true*, if S contains the positive literal edges of T_x, and $\theta(x) = $ *false* otherwise.

Since S is a t–spanner and S contains no clause edge it follows from Lemma 7 that there is at least one adjacent pair of literal edges for every clause edge. Hence, θ satisfies all clauses. □

4 MinS$_t$ for Weighted, Planar Graphs

For the proof of the second part of Theorem 2, we again transform an instance of P3SAT to an instance of MinS$_t$ by extending variable and clause vertices to appropriate components. The assignment of edge weights of value 2 helps lowering the bound on t, thus yielding a stronger result than in the unweighted case. But we cannot expect to reduce the gap by this technique even further.

The variable components are the same with all edges having unit edge weight, and the results about minimum t–spanners for these components remain valid (Lemma 6). The clause components again consist of four clause vertices, but now three sides of the quadrilateral remain unconnected. Only one side is connected by two consecutive forced $(t-1)$–components with unit edge weights. As before, we have one clause edge, now having edge weight 2. We combine the components to form the truth assignment testing components as we did in the unweighted case by identifying the corresponding vertices (see Fig. 3 for an example). Using similar arguments as in Sect. 3, we get an equivalent of Lemma 7 now for all values of $t \geq 3$.

It is easily seen that the constructed graph is again planar and biconnected. By choosing $W = 6m + 36m(t-1)(2t-1) + 2m(t-1)(2t-1)$, the arguments of the previous section can be repeated to complete the proof of part 2 of Theorem 2.

Theorem 2 can be easily generalized to allow rational numbers for $t \geq 3$ by using forced $(\lfloor t \rfloor - 1)$–components in the construction described above. All results about minimum t–spanners then keep valid.

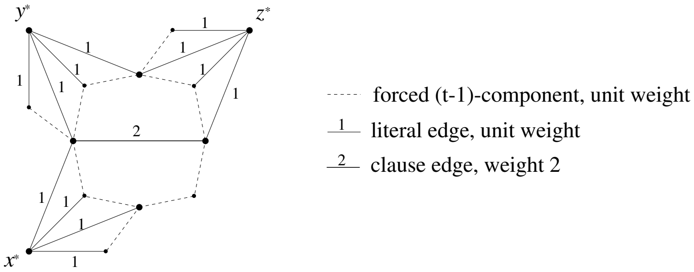

Fig. 3: The truth assignment testing component in the weighted case

5 MinS$_t$ for Planar Digraphs

To show the $\mathcal{NP}$–completeness for digraphs, we again use a modification of the reduction of the previous sections. Here we only show details of the construction for the unweighted case, since the weighted case is then straightforward from what has been established so far.

Forcing Arcs into a Minimum *t*–Spanner. Similar to the undirected case, an arc (a, b) of a digraph can be forced to be in every minimum t–spanner as described in [2]. For this purpose we create two new vertices c and d, add two arcs (c, b) and (d, b), and then add two distinct directed paths of length $t - 1$ from c to a and from d to a, respectively. Then, a minimum t–spanner of this component consists of arc (a, b) and all arcs of the paths of length $t - 1$.

Construction of the Instance. For the construction of the instance see Figure 4. The *variable components* (see Figure 4(a)) again consist of literal and auxiliary vertices, as well as literal and auxiliary arcs. The orientation of these arcs depends on the orientation of the corresponding clause arc. As in the undirected case this construction guarantees that every minimum t–spanner of such a variable component only contains consistent literal arcs (cf. Lemma 6).

The *clause components* are analogous to the undirected case, where the clause arc and the forced $(t-2)$-components are oriented such that they start and end at the same vertices of the quadrilateral. Figure 4(c) shows an example of a directed truth assignment testing component. It is easily seen that the graph is planar and oriented. Choosing $W = 6m + 36m(t-1)(2t-1) + 4m(t-2)(2t-1)$, as in the undirected

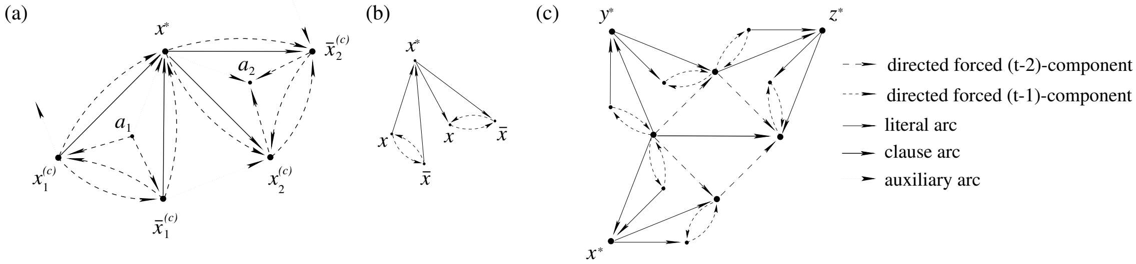

Fig. 4: (a) Part of the variable component for the variable x occurring in clause c, (b) its symbolic representation, and (c) the truth assignment testing component for unweighted digraphs

case, the proof of the equivalence of P3SAT and MinS_t is straightforward as before.

Weighted Digraphs. In the weighted, directed case the same variable components (unit arc weights) are used. The clause components are the ones from the weighted, undirected case, and orientations are determined analogously.

Acknowledgements

We thank two anonymous referees for their helpful suggestions and comments.

References

[1] Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete Comput. Geom.*, 9:81–100, 1993.

[2] Leizhen Cai. $\mathcal{NP}$-completeness of minimum spanner problems. *Discrete Applied Math.*, 48:187–194, 1994.

[3] Leizhen Cai and D.G. Corneil. Tree spanners. *SIAM J. Discrete Math.*, 8(3):359–387, 1995.

[4] S.L. Hakimi and S.-T. Yau. Distance matrix of a graph and its realizability. *Q. J. Mech. appl. Math.*, 22:305–317, 1964.

[5] Guy Kortsarz. On the hardness of approximating spanners. to appear at APPROX'98, 1998.

[6] Sanjeev Khanna and Rajeev Motwani. Towards a syntactic characterization of PTAS. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing, STOC'96*, pages 329–337, 1996.

[7] Guy Kortsarz and David Peleg. Generating sparse 2–spanners. In *Proceedings 3rd Scandinavian Workshop on Algorithm Theory, SWAT'92*, pages 301–309, 1992.

[8] A.L. Liestman and Thomas Shermer. Grid spanners. *Networks*, 23:123–133, 1993.

[9] Anthony Mansfield. Determining the thickness of graphs is $\mathcal{NP}$-hard. *Math. Proc. Camb. Phil. Soc.*, 93:9–23, 1983.

[10] Christos H. Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

[11] David Peleg and Alejandro A. Schäffer. Graph spanners. *J. of Graph Theory*, 13:99–116, 1989.

[12] David Peleg and Jeffrey D. Ullman. An optimal synchronizer for the hypercube. In *Proceedings 1987 6th ACM Symposium on Principles of Dist. Comp., Vancouver*, pages 77–85, 1987.

[13] José Soares. Graph spanners: a survey. *Congressus Numerantium*, 89:225–238, 1992.

[14] G. Venkatesan, Udi Rotics, M.S. Madanlal, Johann A. Makowsky, and C.P̃andu Rangan. Restrictions of minimum spanner problems. *Information and Computation*, 136(2):143–164, August 1997.