Two Structure Theorems on Tree Spanners

by

Ioannis E. Papoutsakis

A thesis submitted in conformity with the requirements for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

© Copyright by Ioannis Papoutsakis January 1999
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-45520-3
Abstract

The concept of a spanner of a graph has received a lot of attention in the last ten years due to its applications in networks, robotics, biology, and theoretical computer science. A $t$-spanner $G'$ of a connected graph $G$ is a subgraph of $G$, such that for every pair of nodes the distance between them in $G'$ is at most $t$ times their distance in $G$. In other words, $G'$ approximates the distances in $G$, using fewer edges.

We are interested in spanners that are trees, because a tree is a minimal connected graph. Given an integer $t$, there are connected graphs that do not admit a tree $t$-spanner, but every connected graph admits a tree $t$-spanner for some $t$. Given a graph $G$ polynomial time algorithms are known that determine whether $G$ has a tree $t$-spanner for $t = 1$ or 2. For $t > 3$ the problem is NP-complete; the $t = 3$ case is still unresolved.

In this thesis we present two new theorems on tree $t$-spanner admissible graphs. First, since a tree does not contain any cycles, given a graph $G$, for every cycle $C$ of $G$, and for every tree $t$-spanner $T$ of $G$, we present an exact lower bound on the number of edges of $C$ that are not in $T$. Second, we present a characterization of tree $t$-spanner admissible graphs in terms of decomposition. This theorem has some interesting algorithmic aspects, and it is shown that it subsumes various known results.
Acknowledgments

During my M.Sc. program, I have received constant encouragement, invaluable guidance and enormous support from my supervisor, Professor Derek Corneil. His insight and criticism provided me with a steady source of inspiration. I am sincerely grateful to him for his advice and friendship.

I thank Professor Michael Molloy for being the second reader of this thesis and for providing helpful suggestions. I also thank Professor Jeremy Spinrad for substantially supporting my admission to the graduate program of the University of Toronto.

I wish to express my gratitude to the members of the department, who make the department a great place to work and study. I also thank the University of Toronto for financial assistance.

Finally, I thank my friends, and my family.
to my teacher in Mathematics

Manolis Maragakis
Contents

List of Figures

1 Introduction

1.1 Statement of the problem ........................................ 4
1.2 Motivation ...................................................... 6
1.3 Basic definitions ............................................... 7
1.4 Previous work .................................................. 10
  1.4.1 Tree spanners .............................................. 10
  1.4.2 Sparse spanners ........................................... 12
  1.4.3 Different approaches .................................... 13
1.5 Overview of the thesis ......................................... 14

2 A necessary condition

2.1 Combining a cycle and a tree spanner .......................... 16
2.2 An application ................................................ 20

3 A necessary and sufficient condition

3.1 An observation .................................................. 22
3.2 A basic Lemma ................................................ 23
  3.2.1 The midst of a path .................................. 23
  3.2.2 A lemma for connecting a cycle and a path .......... 25
3.3 A decomposition theorem on tree spanners .................... 28
  3.3.1 The $t$-star graph ................................... 28
  3.3.2 Tree $t$-spanner vs $t$-star ............................ 30
3.4 Relation to previous work .................................... 33
4 Conclusions and future work. 35
   4.1 Conclusions. .................................. 35
   4.2 The algorithmic aspects of theorem 7. ............... 35
   4.3 Future work ..................................... 36

Bibliography 38

A Appendix 42
# List of Figures

1.1 The set $Y$ of pairs $(m,t)$, such that $G$ admits a corresponding spanner. . . . 6

2.1 A tree chord in a cycle. ...................................................... 18
2.2 Left: a cycle with exactly $\left\lfloor \frac{k-2}{t-1} \right\rfloor$ missing edges. Right: The $t=3$ case. . . . 19
2.3 Left: a cycle of size $k$, and the corresponding path $P$. Right: a cycle and a
    tree path in a tree 3-spanner admissible graph. .......................... 21

3.1 A tree 3-spanner, and a cycle .............................................. 23
3.2 Construction of the new paths $P'_1$, and $P'_2$. ........................... 27
3.3 How are the components of $T \setminus M$ related to the components of $G \setminus W$? . . 28
3.4 Examples of small $t$-stars .................................................... 29
Chapter 1

Introduction

1.1 Statement of the problem

In the second half of the 1980's the concept of spanner first appears:

Given a connected graph $G$, a subgraph $G'$ is a $t$-spanner of $G$ if for every pair of vertices, the distance between them in $G'$ is at most $t$ times their distance in $G$. We refer to $t$ as the stretch factor of $G'$.

The aim is to construct such a subgraph, under the following natural restrictions. First, we want the stretch factor to be as small as possible. Actually, the smaller $t$ is, the more accurately $G'$ describes the distances in $G$. Second, we require $G'$ to be as small as possible. This is a demand for an economical description of $G$ by $G'$. As in most real life problems, there is a tradeoff between these two requirements. Indeed, by deleting an edge from a graph, we cannot decrease the distance between any pair of vertices; at least one pair of vertices will have their distance increased in an unweighted graph. So, if we want to give a precise picture of $G$, using $G'$, we should include in $G'$ almost all the edges of $G$, which implies that $G'$ is not an economical version of $G$. Obviously, similar arguments hold when we try to minimize the number of edges in $G'$.

The definition of a spanner is meaningful for almost\footnote{For example, the case of weighted graphs with negative weights is an exception. This case is also an exceptional one in general for distance problems. Note that the arguments in the previous paragraph do not hold for such graphs.} all known kinds of graphs. For example, $G$ can be a simple, or a weighted, or a directed graph. This “universal” applicability of spanners, makes them a very interesting tool to approximate distances in graphs.
Hence, as we will see in the next section, spanners have many applications in a variety of areas.

Since the universal quantifier on pairs of vertices in the above definition applies to both $G$, and $G'$, $G'$ has to be a spanning subgraph of $G$. We attempt to impart a flavor of the spanner problem with the following example. Let us consider the simplest case of a simple graph $G$ with $n$ vertices, and $M$ edges. Since a spanner of $G$ has to be a spanning subgraph, a reasonable way to measure a spanner's efficiency is the number of its edges.

A key problem concerning spanners is whether there is a spanner of $G$ with stretch factor $t$, and $m$ edges. First, observe that since $G$ is connected, $G'$ has to be connected as well. So, $m \geq n - 1$, since a minimal connected graph is a tree. Second, if $t \geq n$, then every connected subgraph of $G$ is a $t$-spanner of $G$. Third, if we add an edge in a $t$-spanner of $G$, it remains a $t$-spanner, and if there is not a $t$-spanner of $G$ with $m$ edges, then there is not a $t$-spanner of $G$ with fewer than $m$ edges.

The first two observations, together with the fact that a spanner is a subgraph of $G$, and the distances in the spanner cannot be shorter than the distances in $G$, imply the following. The values of $t$, and $m$ that we should be interested in lie in the rectangle $R$: $1 \leq t \leq n - 1$, and $n - 1 \leq m \leq M$. Now, let $Y$ be the set of pairs $(m, t)$, such that $G$ admits a $t$-spanner with $m$ edges, and $N$ the set of the remaining pairs in $R$. The third observation of the previous paragraph implies that there is a line which partitions $R$ into the sets $Y$, and $N$. Also, this line is a function $t_{\min}(x)$, which determines the minimum stretch factor that we can achieve for $G$, using at most $x$ edges. By the same observation we see that $t_{\min}$ is a non-increasing function.

For an arbitrary graph $G$, the corresponding rectangle $R$, and a possible partition of it are shown in figure 1.1. From the graph theoretic point of view, it would be nice to provide more characteristics of this function, in particular the values of $t$, and $m$ at the points where the partitioning line intersects the boundaries of $R$. When $t = 1$ we easily see that a simple graph $G$ admits a 1-spanner if and only if $m = M$. This thesis provides two theorems that support our understanding of what happens on the left vertical boundary of $R$ ($m = n - 1$). In other words, we are interested in tree $t$-spanners.
1.2 Motivation

The notion of spanners first appeared in various forms in the literature. In 1987 Peleg and Ullman proposed that the quality of a spanner for a given simple graph is closely related to the time and communication complexities of any synchronizer for the network based on this graph [28]. In 1986 Bhatt, Chung, Leighton, and Rosenberg [3] seeking optimal solutions of tree machines, considered the dilation of an efficient embedding of a graph in another graph, which is a parameter similar to the stretch factor. The two previous cases refer to unweighted graphs. In 1986 Chew [12] introduced the concept of spanners for complete graphs whose vertices are points in the plane, and the weight of an edge is the distance of its endpoints in some $L_p$ metric. Moreover Dobkin, Friedman, and Supowit [15] studied such graphs in the Euclidean ($L_2$) metric. The conclusion was that there are planar graphs (a planar graph is sparse, compared to the complete graph) that are almost as good as the complete graph.

The demand for understanding in the previous three areas where the concept of a spanner first appeared, motivated various graph theoretical problems, many of which have not yet been resolved. During the 1990's the early concepts gave rise to other applications. For example, in the area of networks, the succinct routes follow the edges of a sparse spanner. So, efficient routing schemes can use a spanner in order to maintain succinct routing tables [29]. Such spanners can also be used as a substitute for the original network while keeping similar communication costs [30, 24, 23, 25]. Moreover, spanners of low degree can be used
for efficient broadcast [21]. In [31] the notion of tree spanners is used to find approximate solutions to the bandwidth minimization problem.

Furthermore, spanners under the constraints of Euclidean distance are used in robotics [14, 19, 20, 22]. In motion planning, when the input of a simple polygon is inaccurate, a special spanner of the visibility graph of the input polygon, called the visibility skeleton, can be used to plan collision free paths inside the real polygon [10]. There have also been spanners for families of graphs used as parallel computation models (hypercubes [28, 23], pyramids [30], multidimensional grids [25], de Bruijn networks [17]). Finally, spanners appear in biology in the process of reconstructing phylogenetic trees from matrices, whose entries represent genetic distances amongst contemporary living species [2].

In some of the above applications of spanners, it is explicitly asked to determine a spanner that is a tree. Of course a tree spanner is always welcome in all the above applications, since a tree is a minimal spanning subgraph. Moreover, the theoretical analysis of tree spanners may provide us with intuition for the general spanner problem, as we saw in the previous section.

1.3 Basic definitions

Most of our terminology follows D. B. West [32]. A simple graph $G$ with $n$ vertices and $m$ edges consists of a vertex set $V(G) = \{v_1, \ldots, v_n\}$ and an edge set $E(G) = \{e_1, \ldots, e_m\}$, where each edge is an unordered pair of vertices. We write $uv$ for the edge $\{u, v\}$. If $uv \in E(G)$, then $u$ and $v$ are adjacent, and $v$ is a neighbor of $u$. The vertices contained in an edge $e$ are its endpoints. Node is a synonym of vertex. The size of a graph is the size of its edge set. Here, $V(G)^2$ is the set of all unordered pairs of vertices of graph $G$. The degree of a vertex is the number of edges containing it. All the nodes have the same degree in a regular graph. Weighted graph means a graph with weights assigned to the edges. A simple directed graph or simple digraph $G$ consists of a vertex set $V(G)$ and an edge set $E(G)$, where each edge is an ordered pair of vertices.

A subgraph of a graph $G$ is a graph $H$ such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$; we write this as $H \subseteq G$ and say that "$G$ contains $H". An induced subgraph of $G$ is a subgraph $H$ such that every edge of $G$ with endpoints contained in $V(H)$ belongs to $E(H)$. If $H$ is an induced subgraph of $G$ with vertex set $S$, then we write $H = G[S]$ and say that $H$ is the
A subgraph of $G$ "induced by $S$".

A walk of length $k$ is a sequence $v_0, e_1, v_1, e_2, \ldots, v_k$ of vertices and edges such that $e_i = v_{i-1}v_i$ for all $i$. A trail is a walk with no repeated edge. A path is a walk with no repeated vertex. A $u, v$-walk $P$ can be written as $P(u, v)$. A cycle is a closed trail of length at least one in which "first=last" is the only vertex repetition. Trails, paths, and cycles are walks and hence have lengths. We write $|C|$ or $|P|$ for the length of a cycle $C$ or a path $P$, respectively. The circumference of a graph is the length of its longest cycle.

A graph $G$ is connected if it has a $u, v$-path for each pair $u, v \in V(G)$ (otherwise it is disconnected). If $G$ has a $u, v$-path, then $u$ is connected to $v$ in $G$. The components of a graph $G$ are its maximal connected subgraphs. A cut-edge or a cut-vertex of a graph is an edge or vertex whose deletion increases the number of components. We use articulation point to mean cut-vertex. A block of a graph $G$ is a maximal connected subgraph of $G$ that has no articulation point. Here, $G_1 \setminus G_2$ is $G_1[V(G_1) \setminus V(G_2)]$. A separating set or vertex cut of a graph $G$ is a set $S \subseteq V(G)$ such that $G \setminus S$ has more than one component. A graph $G$ is $k$-connected if every vertex cut has at least $k$ vertices. As a generalization of vertex cut, a cut-set is a set of vertices and edges (or in other words a graph) whose deletion increases the number of components. A graph $G$ is an edge bonding of two graphs $G_1$ and $G_2$ if $G = G_1 \cup G_2$, and $G_1$ and $G_2$ have exactly one edge in common.

A complete graph or clique is a simple graph in which every pair of vertices forms an edge. An independent set in a graph $G$ is a vertex subset $S \subseteq V(G)$ such that the induced subgraph $G[S]$ has no edges. The complement of a simple graph $G$, written $\bar{G}$, is a graph having the same vertex set as $G$, such that $u, v$ are adjacent in $\bar{G}$ if and only if $u, v$ are not adjacent in $G$. A graph is bipartite if its vertex set can be partitioned into two independent sets. A complete bipartite graph is a bipartite graph in which the edge set consists of all pairs having a vertex from each of the two independent sets in the vertex partition.

We use $K_n$, $P_n$, $C_n$, respectively, to denote any graph that is a clique, path or cycle with $n$ vertices. Similarly, we use $K_{r,s}$ to denote any complete bipartite graph with partite sets of sizes $r$, and $s$. In particular, $K_{1,n}$ is called a star. A cut-set $S$ that is a star is called a star cut. Note that the induced subgraph of such a cut-set $S$ of a graph $G$ may not be a $K_{1,n}$, but $G[S]$ has to contain a spanning $K_{1,n}$. A vertex in a graph $G$ is universal if it is
adjacent to all other vertices of this graph. An edge is *dominating* if all the vertices in the graph are adjacent to at least one of its endpoints.

As pointed out in the previous section trees play an important role in the study of spanners. The following quote from [32] provides the definitions regarding trees needed for this thesis.

The word “tree” suggests branching out from a root and never completing a cycle. Trees as graphs have many applications, especially in data storage and communications (including computation of distances).

A graph having no cycle is *acyclic*. A *forest* is an acyclic graph; a *tree* is a connected acyclic graph. A leaf (or pendant vertex) is a vertex of degree 1. A *spanning subgraph* of $G$ is a subgraph with vertex set $V(G)$. A *spanning tree* is a spanning subgraph that is a tree.

If $G$ has a $u,v$-path, then the *distance* from $u$ to $v$, written $d_G(u,v)$ or simply $d(u,v)$, is the least length of a $u,v$-path. If $G$ has no such path, then $d(u,v) = \infty$.

Moreover, the *diameter* of a graph $G$ is $\max_{u,v \in V(G)} d(u,v)$. Definitions which have been invented for the needs of this thesis will be presented in the body of the thesis. Results on various restricted families of graphs will also be presented. In order to avoid cluttering the thesis with their definitions, these terms, written in italics, are given in the appendix. All graphs in this thesis, unless otherwise specified, are simple, unweighted, undirected, and connected.

As an aside to these definitions, we present a basic observation that simplifies the definition of a spanner of a graph, since it allows us to consider only adjacent vertices of the graph. The following proposition appears in [9] in a more general form.

**Proposition 1** ([9]) A spanning subgraph of an unweighted simple graph $G$ is a $t$-spanner of $G$ if and only if for every edge $xy$ in $G$: $d_H(x,y) \leq t$.

*Proof*. If $H$ is a $t$-spanner, then for every pair of vertices $x, y$ in $G$, $d_H(x,y) \leq td_G(x,y)$. If $x$ and $y$ are connected in $G$ then $d_G(x,y) = 1$, so for every edge $xy$ in $G$: $d_H(x,y) \leq t$. For sufficiency, it suffices to show that $d_H(x,y) \leq td_G(x,y)$ for two arbitrary vertices $x, y$ of $G$. Let $P$ be a shortest path between $x$, and $y$ in $G$. Then for each edge $uv$ on $P$, $d_H(u,v) \leq t$. Therefore, $d_H(x,y) \leq \sum_{uv \in P} d_H(u,v) \leq \sum_{uv \in P} t = t|P| = td_G(x,y)$. \qed

9
1.4 Previous work

A graph theoretical study of spanners started in 1989 by Peleg and Schäfer [27]. In this paper the following theorem appears for connected unweighted simple graphs.

**Theorem 1 ([27])** For every $n$-vertex graph $G$ and for every $1 < t < n$, there exists a (polynomial-time-constructible) $(4 \log n + 1)$-spanner with at most $tn$ edges.

This theorem provides us with a strictly determined region of rectangle $R$ of figure 1.1 which is a subset of set $Y$. It is actually a lower bound of set $Y$. In the same paper, the authors consider the intractability of the general spanner problem. In particular the following theorem is presented.

**Theorem 2 ([27])** The problem of determining, for a given graph $G = (V, E)$ and two integers $t, m \geq 1$, whether $G$ has a $t$-spanner with $m$ or fewer edges, is NP-complete.

These are the two basic guidelines in attacking the spanner problem. The first is graph theoretic, and the second algorithmic. The rest of this section is partitioned into three subsections. The first considers tree spanners of graphs, and is the most important for this thesis. The second speaks of sparse spanners, and the last involves different approaches to the spanner problem.

### 1.4.1 Tree spanners

A formal definition of the tree $t$-spanner decision problem is: Given a simple graph $G$, is there a tree $T \subseteq G$ such that $d_T(u, v) \leq t \cdot d_G(u, v)$ for every $u, v \in G$? Note that in this version of the problem $t$ is not in the input of the problem. We examine how the difficulty of the problem changes with $t$. If $t = 1$, then all the edges of $G$ have to be in the 1-spanner, when $G$ is unweighted. Hence, an unweighted graph is tree 1-spanner admissible if and only if it is a tree. When $G$ is a weighted graph, then a tree 1-spanner, if it exists, is a minimum spanning tree, and can be found in polynomial time [9]. If $t > 1$ the notion of tree $t$-spanner admissible graphs becomes more fruitful.

As mentioned in theorem 2, the general spanner problem is NP-complete. Note that in this case the stretch factor $t$ is part of the input of the problem. When this general problem is restricted to tree spanners ($m = n - 1$), it remains hard. The tree $t$-spanner problem on weighted graphs is shown to be NP-complete for any fixed rational $t > 1$ in [9].
graph is unweighted it suffices to consider only integral values of $t$. The NP-completeness of the tree $t$-spanner problem for fixed integer $t \geq 4$ is established in [9] for unweighted graphs. The complexity status of the case $t = 3$ is unknown.

**Theorem 3 ([9])** For any fixed $t \geq 4$, the tree $t$-spanner problem is NP-complete.

In the same paper, a linear algorithm is presented for recognizing tree 2-spanner admissible unweighted graphs. This particular algorithm is based on a characterization of tree 2-spanner admissible graphs in terms of decomposition. It is remarkable that a tree 2-spanner of a graph coincides with a tritree\(^2\) of a graph, a concept introduced by Bondy [4] in his work on cycle double covers. The family of tree 2-spanner admissible graphs is characterized by the following theorem:

**Theorem 4 ([4])** A graph $G$ is tree 2-spanner admissible if and only if either

1. it has a universal vertex, or
2. every block of it is tree 2-spanner admissible, or
3. it is an edge bonding (on edge $e$) of two tree 2-spanner admissible graphs where $e$ is a tree edge in both graphs.

The new concept of the skeleton tree is introduced in [9], which provides a much clearer picture of tree 2-spanner admissible graphs than theorem 4. The set of edges that belong to every tree 2-spanner of a graph form the skeleton of the graph, which is proven to be a tree. From the skeleton tree of a graph, all the possible tree 2-spanners of that graph can be easily composed.

To the contrary, for the tree 3-spanner admissible graphs it is not known if they admit such an interesting structure as the skeleton tree. As mentioned the complexity status of the tree 3-spanner problem is unknown. In [7] there is the following result on tree 3-spanner admissible graphs, which reveals a sort of complementary relationship between star cuts and dominating edges. Recall that an edge $e$ of $G$ is a dominating edge if every vertex of $G$ is adjacent to at least one end of $e$, and note that a star cut $S$ of $G$ is a vertex cut such that $G[S]$ has a spanning star.

\(^2\)Recall that terms written in italics are either defined immediately or appear in the Appendix.
Theorem 5 ([7]) Let $G$ be a graph containing no star cut. Then it has a tree 3-spanner if and only if it contains a dominating edge.

In [7] it is also observed that cographs, split graphs, and complement of bipartite graphs always admit a tree 3-spanner. In [26] it is proved that all interval, and permutation graphs admit a tree 3-spanner. Furthermore, the tree 3-spanner problem is solved for regular bipartite graphs in [26]; a regular bipartite graph is tree 3-spanner admissible if and only if it is complete. This theorem has as immediate corollary that for $k \geq 3$ all $k$-cubes are not tree 3-spanner admissible graphs. In [31] it is proved that all convex bipartite graphs have a tree 3-spanner, and such a spanner can be constructed in linear time. In [5] is shown that every strongly chordal graph admits a tree 4-spanner.

In [16] tree spanners in planar unweighted graphs are considered. In particular, they present a polynomial algorithm that decides for planar unweighted graphs with bounded face length whether there is a tree $t$-spanner, for any fixed $t$. Furthermore, in the same paper it is proved that it can be decided in polynomial time whether a planar unweighted graph has a tree 3-spanner. The method used in this paper makes strong use of the planarity. They also present a lemma that characterizes a planar tree $t$-spanner admissible graph $G$ in terms of decomposition of the dual graph of $G$. These algorithms are based on dynamic programming.

Finally, the tree $t$-spanner problem is studied for digraphs in [9]. It seems that tree spanner problems on digraphs are at least as hard as tree spanner problems on undirected graphs. Surprisingly though, an $O((m + n)\alpha(m,n))$ algorithm is provided for finding a minimum (with respect to $t$) tree $t$-spanner of a digraph with $n$ vertices, and $m$ edges, where $\alpha(m,n)$ is a functional inverse of Ackerman's function.

### 1.4.2 Sparse spanners

Assume that we are given a graph $G$, and we are asked to find a $t$-spanner of $G$ with the minimum number of edges. If $G$ admits a tree $t$-spanner, then this problem is reduced to the tree $t$-spanner problem. Otherwise, we have to determine a “sparsest” subgraph of $G$ that approximates the distances in $G$. This more general version of the tree $t$-spanner problem is called the minimum $t$-spanner problem.

In [1] a simple algorithm is given for constructing sparse spanners for arbitrary weighted graphs. This particular algorithm is similar to Kruskal's algorithm (see [32]) for minimum
spanning trees. The main difference is that we are allowed to make a cycle by adding an edge to the spanner. The weight of such an edge has to be much smaller than the weight of the cycle formed by the addition of this edge. This algorithm provides a constructive proof of a theorem similar to theorem 1, which applies to weighted graphs as well. The main strategy for this theorem is that of dividing the graph into planar components, and the authors believe that this strategy cannot be extended to yield the optimal answer. The same algorithm is used to obtain specific results for planar graphs and Euclidean graphs.

Finding a minimum \( t \)-spanner is NP-hard for unweighted graphs, where \( t \) is a fixed integer, and \( t \geq 2 \) [27, 8]. Moreover in [11] the minimum \( t \)-spanner problem, for each fixed \( t \geq 2 \), remains NP-hard on graphs with maximum degree greater than or equal to 9. In the same paper it is also shown that for graphs with maximum degree less than or equal to 4, the minimum 2-spanner problem can be solved by a polynomial time algorithm. For the remaining combinations of \( t \), and the maximum degree of a graph, the complexity status of the minimum \( t \)-spanner problem remains open. The complexity of this problem is studied for various restricted families of graphs in [31].

The notion of tree spanners is extended to \( k \)-tree spanners in [7]. So, in this case we are interested in a special family of sparse spanners. A basic characteristic of \( k \)-trees is that they are \( k \)-connected, and therefore they are important for the study of fault tolerant networks. Unfortunately, it is shown in [7] that the \( k \)-tree \( t \)-spanner problem is NP-complete for all integers \( k, t \geq 2 \), except \( k = t = 2 \) which is unknown. In the same paper it is shown that for cographs a 2-tree \( t \)-spanner can be found in linear time \( (t \geq 3) \), while the 2-tree 2-spanner problem is still open for cographs.

1.4.3 Different approaches

According to our definition of a spanner, the distances in the spanner are at most \( t \) times the distances in the corresponding graph. In [24] the notion of additive graph spanners is introduced. An additive \( t \)-spanner of a graph \( G \) is a spanning subgraph \( H \) of \( G \) in which \( d_H(x, y) \leq d_G(x, y) + t \) for every pair of vertices \( x, y \) of \( G \). In [5] additive tree spanners for chordal graphs are studied. In [7] there is a more general definition of tree spanners, which considers the idea of additive spanners, and distinguishes the roles of adjacent and non-adjacent vertices: a tree \((s, t)\)-spanner of a graph \( G \) is a spanning tree \( T \) such that for any two vertices \( x \) and \( y \), \( d_T(x, y) \leq t \) for \( xy \in E(G) \), and \( d_T(x, y) \leq d_G(x, y) + s \).
otherwise. Note that an additive t-spanner coincides with a \((t, t + 1)\)-spanner. In [7] there is also an extensive study of a special type of t-spanner, the \(\beta_t\)-tree, which is closely related to additive tree spanners. A \(\beta_t\)-tree \(T\) of a weighted graph \(G = (V, E, w)\) is a spanning tree such that for every edge \(xy\) of \(G\), \(d_T(x, y) \leq w(xy) + t\).

Throughout the previous subsection, we considered spanners with the minimum number of edges. However, another significant parameter when selecting a good spanner, is the maximum degree of the spanner. For instance, in many applications of spanners in networks, the degree of each node in the spanner directly translates into memory requirements at the node, and high local degrees mean higher workload on the involved nodes. In [21] there is an extensive study of the low-degree 2-spanner problem.

In a different approach consider the following problem in network design: given an undirected graph with non-negative weights on the edges, the goal is to find a spanning tree such that the average distance between any pair of vertices in the tree is minimized. This problem is known as the Network Design Problem. It is obvious that in a tree t-spanner of a graph the sum of distances between every pair of nodes in the tree is at most \(t\) times (there can be a better upper bound) the sum of distances in the graph. This problem is shown to be NP-complete in [18], and in [33] a polynomial time approximation scheme for this problem is presented.

Another approach is to consider a spanner of a graph \(G\) that is not necessarily a subgraph of \(G\). The constraint for a spanner to be a subgraph is replaced by the restriction that distances in the spanner cannot be smaller than distances in the graph. In this case the definition of a t-spanner is the following: a t-spanner of a graph \(G\) is a set of weighted edges on the vertices of \(G\) such that the distances in the spanner are not smaller and within a factor of \(t\) from the corresponding distances in \(G\). Fast algorithms for constructing such \(t\)-spanners are presented in [13].

1.5 Overview of the thesis

A tree does not contain any cycles. A tree spanner admissible graph that is not a tree contains some cycles. How many edges should we remove from a cycle in order to get a tree spanner? In chapter 2 we present a strict lower bound of the number of such edges. In an application of this theorem we see how the stretch factor \(t\) effects the length of tree t-
spanner paths between nodes of a cycle. This application provides an interesting indication of the difficulty of the tree 3-spanner problem, whose complexity status is unknown.

In the next chapter, we see that the cycles of a graph have to follow the branches of a tree $t$-spanner of the graph. This observation is presented formally as a lemma, where we introduce the midst of a path. This lemma describes a set of nodes in a tree $t$-spanner admissible graph, which is a cut set of the graph. The induced subgraph of such a separating set is a $t$-star. This new family of graphs can be recognized in polynomial time for every $t$, and every connected graph is a $t$-star for some $t$. This approach to the tree $t$-spanner problem ends with a characterization of tree $t$-spanner admissible graphs in terms of decomposition.

In the last chapter we discuss the algorithmic aspects of this thesis, and present future work.
Chapter 2

A necessary condition

In this chapter, we introduce a necessary condition that every cycle of a tree $t$-spanner admissible graph has to satisfy. A tree $t$-spanner $T$ of a graph $G$ does not contain any cycles, so if there is a cycle $C$ in $G$, then some edges of $C$ should not be in $T$. How many of these edges are there? We prove an exact lower bound of the number of such edges. This necessary condition leads to an application, which demonstrates a different situation when $t = 2$ (and we know a polynomial time algorithm), and $t > 2$, where the problem is either NP-complete ($t > 3$) or unresolved ($t = 3$).

2.1 Combining a cycle and a tree spanner

We determine a lower bound of the number of edges of a cycle that are not in a tree spanner. First, we give a formal definition of this quantity.

**Definition 1** Let $G$ be a graph, $C$ a cycle of $G$, and $T$ a tree $t$-spanner of $G$. Then, $e(C, T) = |E(C) \setminus E(T)|$ is the number of edges of $C$ missing from $T$.

If a cycle of a tree $t$-spanner admissible graph is big enough compared to the stretch factor $t$, then there is at least one path in the tree $t$-spanner, which is like a chord of that cycle. This is the idea of the following lemma.

**Lemma 1** Let $G$ be a graph, and $T$ a tree $t$-spanner of $G$. If $C$ is a cycle in $G$, and $|C| \geq t + 2$, then there exists a path $P(u, v) \subseteq T$, such that $V(P) \cap V(C) = \{u, v\}$, and $E(P) \cap E(C) = \emptyset$. 

16
Proof. Since $C$ is a cycle, there should be an edge of $C$, say $pq$, missing from $T$. Since $T$ is a $t$-spanner of $G$, there is a path $P_0(p, q) \subseteq T$, such that $|P_0| \leq t$. Since $|C| \geq t + 2$, the distance between $p, q$ in $C \setminus \{pq\}$ is greater than $t$. So $P_0$ cannot be completely in $C$, and since $p, q \in C$, there is a path $P(u, v) \subseteq P_0 \subseteq T$, preserving the properties described by the lemma. $\square$

In general, when we want to construct a spanning tree of a graph $G$, it suffices to remove at least one edge from every cycle of $G$. This is not enough if we want to find a spanning tree which is a spanner as well. The following theorem, the first of the two major contributions of the thesis, provides us with a lower bound for the number of missing edges from a cycle $C$.

**Theorem 6** Let $G$ be a graph. For every tree $t$-spanner $T$ of $G$, and for every cycle $C$ of $G$:

$$e(C, T) \geq \left\lceil \frac{|C| - 2}{t - 1} \right\rceil.$$  

**Proof.** We will prove the following proposition, which easily implies the theorem.

Let $G$ be a graph. For every tree $t$-spanner $T$ of $G$, and for every cycle $C$ of $G$, such that $|C| \geq l \geq 3$:

$$e(C, T) \geq \left\lceil \frac{l - 2}{t - 1} \right\rceil.$$  

Consider a graph $G$, and let $T$ be an arbitrary tree $t$-spanner of $G$. From now on, we will not refer to any other tree $t$-spanner, so instead of $e(C, T)$, we will write just $e(C)$. We prove the proposition using induction on $l$.

**Basis:** $3 \leq l \leq t + 1$. Then $\left\lceil \frac{l-2}{t-1} \right\rceil = 1$. So, we have to prove that for every cycle $C \subseteq G$, with $|C| \geq 3$ there will be at least one missing edge of $C$ from $T$. This holds, because otherwise $C \subseteq T$, which is a contradiction, since $T$ is a tree.

**Induction hypothesis:** We assume that the proposition holds for $l \leq k$ (strong induction). Note that this implies that for every cycle $C$:

- $e(C) \geq \left\lceil \frac{k-2}{t-1} \right\rceil$, when $|C| \geq k$, and
- $e(C) \geq \left\lceil \frac{|C|-2}{t-1} \right\rceil$, when $|C| < k$.

**Induction step:** We will prove that the proposition holds for $l = k + 1 > t + 1$. Consider a cycle $C \subseteq G$, with $|C| \geq k + 1 \geq t + 2$. From lemma 1 we know that there should be
a path \(P(u,v) \subseteq T \subseteq G\) such that \(V(P) \cap V(C) = \{u,v\}\), and \(E(P) \cap E(C) = \emptyset\). The situation is shown in figure 2.1:

![Figure 2.1: A tree chord in a cycle.](image)

Since \(C\) is a cycle, there are two paths \(D_1\), and \(D_2\) that connect \(u\), and \(v\) in \(C\). These two paths, and path \(P\) determine two cycles: \(C_1 = D_1 \cup P\), and \(C_2 = D_2 \cup P\). Let \(d_1\) be the length of path \(D_1\), and \(d_2\) the length of path \(D_2\). Since \(|P| \geq 1\), we see that \(|C_1| \geq d_1 + 1\), and \(|C_2| \geq d_2 + 1\). Consider now the following cases:

- \(|C_1| \geq k\) or \(|C_2| \geq k\). Without loss of generality, assume that \(|C_1| \geq k\). By the induction hypothesis we know that \(e(C_1) \geq \left\lfloor \frac{k-2}{t-1} \right\rfloor\), since \(|C_1| \geq k\). Moreover, \(e(C_2) \geq 1\), since \(C_2\) is a cycle. Since \(P \subseteq T\), all the missing edges of \(C_1\), and \(C_2\) are edges of \(C\). So:

\[
e(C) = e(C_1) + e(C_2) \geq \left\lfloor \frac{k-2}{t-1} \right\rfloor + 1 \geq \left\lfloor \frac{(k+1)-2}{t-1} \right\rfloor = \left\lfloor \frac{l-2}{t-1} \right\rfloor
\]

- \(|C_1| < k\) and \(|C_2| < k\). In this case we use the strong induction hypothesis:

\[
e(C_1) \geq \left\lfloor \frac{|C_1| - 2}{t-1} \right\rfloor \geq \left\lfloor \frac{(d_1 + 1) - 2}{t-1} \right\rfloor = \left\lfloor \frac{d_1 - 1}{t-1} \right\rfloor
\]

Similarly for \(C_2\):

\[
e(C_2) \geq \left\lfloor \frac{|C_2| - 2}{t-1} \right\rfloor \geq \left\lfloor \frac{(d_2 + 1) - 2}{t-1} \right\rfloor = \left\lfloor \frac{d_2 - 1}{t-1} \right\rfloor
\]

As in the previous case, since \(P \subseteq T\), all the missing edges of \(C_1, C_2\) are edges of \(C\). So:

\[
e(C) = e(C_1) + e(C_2) \geq \left\lfloor \frac{d_1 - 1}{t-1} \right\rfloor + \left\lfloor \frac{d_2 - 1}{t-1} \right\rfloor \geq \left\lfloor \frac{d_1 + d_2 - 2}{t-1} \right\rfloor
\]
But $d_1 + d_2 = |C| \geq k + 1$, so:

$$e(C) \geq \left\lfloor \frac{(k + 1) - 2}{t - 1} \right\rfloor = \left\lfloor \frac{n - 2}{t - 1} \right\rfloor.$$ 

There are not any other cases to examine, and in both cases the induction step holds.\(\square\)

The lower bound posed by the previous theorem is exact, since for every $k \geq 3$, and for every stretch factor $t$, there is a graph that contains a cycle $C$ of length $k$, such that $e(C) = \left\lfloor \frac{k - 2}{t - 1} \right\rfloor$. For arbitrary $k$ and $t$, a tree $t$-spanner admissible graph $G$ with the previous property is shown on the left hand side of figure 2.2. All the nodes shown in this figure form a cycle $\{v_1, v_2, \ldots, v_k\}$ in the graph. In the figure, only the tree edges appear. Note that the dashed lines between $v_2$ and $v_3$, for example, indicate the tree path $v_2, v_3, \ldots, v_k$.

Consider all the nodes of the cycle except nodes $v_1$, and $v_k$. As we traverse the cycle along the remaining $k - 2$ nodes, there is a missing edge of the cycle per $t - 1$ nodes. Of course the tree in the figure is a $t$-spanner of $G$, since $G$ contains only the edges of the cycle, and the other edges adjacent to node $v_1$, as shown in the figure. On the right hand side of the same figure we see such a graph for the $t = 3$ case, where the dashed lines now are the edges of the graph, that are not in the tree 3-spanner.

Figure 2.2: Left: a cycle with exactly $\left\lfloor \frac{k - 2}{t - 1} \right\rfloor$ missing edges. Right: The $t = 3$ case.

Note that there is a $t - 1$ in the denominator of the lower bound. So, this does not hold for $t = 1$. But, trees are the only tree 1-spanner admissible graphs, and trees do not contain any cycles. Moreover, lemma 2, which we will state and prove in the next chapter, can provide us with an appropriate path $P$ for the proof of theorem 6, such that both cycles
$C_1$, $C_2$ have size less than the size of $C$. This will make the proof of theorem 6 simpler, since we would have to consider only one case in the proof of the induction step. We do not have a very simple proof of lemma 2, though.

2.2 An application

In this section, we apply the previous theorem. We use the obvious fact that for every cycle the number of missing edges can be at most equal to its length. This will help us to determine an upper bound for the length of a tree path that connects two nodes of a cycle.

Consider a tree $t$-spanner admissible graph $G$. Arbitrarily pick a cycle $C$ in $G$. There can be many tree $t$-spanners for $G$. In each of these trees there is a path between every pair of nodes of $C$. We are interested in paths between two nodes of the cycle that do not contain any other node of the cycle. So, consider a tree $t$-spanner $T$ of $G$, and a path $P(u,v) \subseteq T$ between two nodes of $C$, such that $P \cap C = \{u, v\}$.

Between $u$, and $v$ there are two node independent paths in $C$. As in the proof of theorem 6, we name these paths $D_1$, and $D_2$. These two paths, together with $P$ form two cycles $C_1$, and $C_2$, which are different than $C$. For each of these cycles we apply theorem 6. Since $P \subseteq T$, we see that $e(C_1) \leq |D_1|$, and $e(C_2) \leq |D_2|$. Moreover, $|C_1| = |D_1| + |P|$, and $|C_2| = |D_2| + |P|$. So:

$$|D_1| \geq \frac{|D_1| + |P| - 2}{t - 1}, \quad |D_2| \geq \frac{|D_2| + |P| - 2}{t - 1}$$

We sum the left and right hand sides of the above inequalities. We also know that $|C| = |D_1| + |D_2|$, so:

$$|C| \geq \frac{|C| + 2|P| - 4}{t - 1}$$
Hence, for the length of path $P$, we conclude the following corollary:

**Corollary 1** Let $C$ be a cycle in a tree $t$-spanner admissible graph $G$. For every tree $t$-spanner $T$ of $G$, and for every path $P(u, v)$, such that $P \cap C = \{u, v\}$, and $P \subseteq T$:

$$|P| \leq \frac{t-2}{2}|C| + 2.$$

This upper bound is almost exact. For every $t \geq 3$ there exists a graph, which admits a tree $t$-spanner that contains such a path $P$, and $P$ has length very close to that upper bound. The graph on the left hand side of figure 2.3 illustrates a general example for every $t$, and for every cycle length $k$. The nodes $u$, $v$, $p_i \ (1 \leq i \leq k - 2)$ form the cycle $C$. The edges of the tree $t$-spanner form a path $P$, of length $\frac{t-2}{2}(k-1)+2$, from $u$ to $v$ together with edges from each $p_i$ to this path. Here, $u$, $v$ are antipodal nodes of the cycle. In particular $p_i$ is adjacent to $q_i$ on the path $P$. The length of the path $P$ between $q_i$, $q_{i+2}$ is $t-2$. Note that none of the edges of $C$ is in $T$. On the right hand side of the same figure, we see such a graph for the $t = 3$ case. Note that in this case $t - 2 = 1$ so $q_i$, and $q_{i+1}$ coincide.

Figure 2.3: Left: a cycle of size $k$, and the corresponding path $P$. Right: a cycle and a tree path in a tree 3-spanner admissible graph.

The formula in the previous corollary provides us with an interesting piece of information. If $t = 2$, the coefficient of $|C|$ becomes 0, so the length of $P$ is at most a constant (i.e. 2), and does not depend on the size of the cycle. If $t \geq 3$, the length of $P$, depends on the size of the cycle, and as shown in the previous example, there are cases where this length is not bounded by a constant. So, this might explain the difficulty of the tree $t$-spanner problem, when $t \geq 3$. 

21
Chapter 3

A necessary and sufficient condition

In this chapter we present a structure theorem for tree $t$-spanner admissible graphs, where $t$ is an arbitrary constant. We introduce a new family of graphs, the $t$-stars, which is fundamental for this theorem. Moreover, a few lemmata are presented before the theorem. Finally, we show how this theorem subsumes previous work on tree spanners. This chapter starts with an observation on the relation between a cycle of a graph, and its tree spanner.

3.1 An observation

Consider a tree $t$-spanner admissible graph, and a tree $t$-spanner of this graph. For example, in figure 3.1, we see part of a tree 3-spanner admissible graph $G$. Actually, we see the edges of a tree 3-spanner $T$ of $G$ and the edges of a cycle $C$ in $G$. First, an edge of $C$ cannot connect nodes whose tree distance is greater than 3. Thus while traversing $C$, we cannot jump to an arbitrary node of $G$. Second, one of the basic characteristics of a cycle is that we return back to the node from which we started. As we will see these two facts "imply" that there should be two nodes of $C$, whose distance in $C$ is big, but whose distance in $T$ is small. For example consider vertices $u, v$ in figure 3.1. Another way to pose this observation is that a cycle of $G$ has to follow somehow the branches of $T$. 
3.2 A basic Lemma

In this section, we will formalize some characteristics of a tree $t$-spanner admissible graph, according to the previous observation. Essentially, we continue dealing with a connection between the cycles of a graph, and the paths of its tree spanner. Of course, this connection has to do with the distances in a graph. The definitions in the next section, and the following lemma, are formalizations of the previous observation.

3.2.1 The midst of a path

For the needs of the lemma, we define the distance between a pair of nodes and a node. Consider a pair of adjacent nodes $x, y$ in a graph $G$. A path $P(x, y, u)$ between node $u \in G$ and the pair $\{x, y\}$ is a path from $u$ to one of the nodes $x, y$. If there is such a path in $G$, then the distance from $u$ to $\{x, y\}$, written $d_G(\{x, y\}, u)$, is the least length of a path between $u$ and $\{x, y\}$. Obviously, such a shortest path contains only one of the nodes $x, y$.

Now, consider a path in a graph. We are interested in defining its middle, with respect
to a positive number $t$. Instead of middle, we call this the $t$-midst of a path.

**Definition 2** Consider a path $P(u,v)$. A $t$-midst of $P$, written $M(P,t)$, is a subgraph of $P$ consisting of

- a node $x \in P$, such that: $d_{P}(x,v) > \lceil \frac{t-1}{2} \rceil$, and $d_{P}(x,u) > \lceil \frac{t-1}{2} \rceil$, when $t$ is odd, or

- a pair of nodes $x,y \in P$, and the edge $xy \in P$, such that: $d_{P}(\{x,y\},v) > \lceil \frac{t-1}{2} \rceil$, and $d_{P}(\{x,y\},u) > \lceil \frac{t-1}{2} \rceil$, when $t$ is even.

The proposition below, provides us with an easy necessary and sufficient condition in order for a path to have a $t$-midst.

**Proposition 2** Consider a path $P(u,v)$, and an integer $t \geq 1$. Then, $P$ has a $t$-midst, if and only if $|P| > t$.

**Proof.** (Sufficiency). Consider a path $P(u,v)$ with $|P| > t$. Starting from $u$ we traverse $P$ towards node $v$. As we proceed the distance from $u$ is increasing. We stop at the first node $x$ of $P$, for which $d_{P}(u,x) = \lceil \frac{t-1}{2} \rceil + 1$. There is always such a node $x$, since $P$ is long enough. We consider two cases:

- $t$ is odd. Then, let $M(P,t)$ be node $x$. Since $|P| = d_{P}(u,x) + d_{P}(x,v)$, $d_{P}(x,v) > t - (\lceil \frac{t-1}{2} \rceil + 1) = \lceil \frac{t-1}{2} \rceil$, since $t$ is odd.

- $t$ is even. Let $y$ be the neighbor of $x$ on $P$ in the $v$ direction. Then, let $M(P,t)$ be the edge $\{xy\}$. Now $|P| = d_{P}(u,x) + 1 + d_{P}(y,v)$, and we observe that $d_{P}(u,x) = d_{P}(u,\{x,y\})$, and $d_{P}(y,v) = d_{P}(\{x,y\},v)$. So, $d_{P}(\{x,y\},v) > t - 1 - (\lceil \frac{t-1}{2} \rceil + 1) = \lceil \frac{t-1}{2} \rceil$, since $t$ is even.

In both cases, we have constructed a $t$-midst of $P$.

(Necessity). We observe that if $M$ is a $t$-midst of the path $P(u,v)$, then $|P| = d_{P}(u,M) + |M| + d_{P}(M,v)$. Substituting the distances from the definition of a $t$-midst, we see that $|P| > t$. □

There can be many $t$-midsts in a path of length $t+1$ or more, but at least one. Actually, if $|P| = t+1$, then $P$ has a unique $t$-midst, either a node or a pair of adjacent nodes. Clearly for any $t$-midst $M$, $|M| = |E(M)| = (t+1) \text{ mod } 2$.
3.2.2 A lemma for connecting a cycle and a path

The following lemma, makes use of the definition of a t-midst, in order to connect a cycle of a graph $G$, and a path of a tree t-spanner of $G$. We consider two paths $P_1$ and $P_2$ in $G$ that form a cycle. The first path, $P_1$, is contained in the tree t-spanner, and it is sufficiently long. This lemma provides a constraint about the other path $P_2$. We will show that $P_2$ has to pass "near" a t-midst of $P_1$.

**Lemma 2** Let $G$ be a graph, and $T$ a tree t-spanner of $G$. Consider a cycle $C$ of $G$ composed of two paths $P_1(u,v)$, and $P_2(u,v)$, such that:

1. $V(P_1) \cap V(P_2) = \{u,v\}$,
2. $|P_1| > t$,
3. $P_1 \subseteq T$, and
4. $M$ is a t-midst of $P_1$.

Then there is a node $w \in P_2$, such that: $d_T(M,w) \leq \lfloor \frac{t-1}{2} \rfloor$.

**Proof.** Let $H(P_1, P_2)$ indicate that $P_1$, and $P_2$ satisfy assumptions 1-4 of the lemma. Let $C(M, P_2)$ be the following proposition: There exists a node $w \in P_2$, such that: $d_T(M,w) \leq \lfloor \frac{t-1}{2} \rfloor$. Now, the lemma actually says¹: $H(P_1, P_2) \Rightarrow C(M,P_2)$. We prove the lemma by contradiction, i.e. assume that $C(M,P_2)$ does not hold. Using the assumptions $H(P_1, P_2)$, and $\neg C(M,P_2)$, we construct a sequence of pairs of paths $(P_1^k, P_2^k)$, which preserve the assumptions of the lemma, i.e. $H(P_1^k, P_2^k)$ holds. Eventually, we reach a contradiction.

First we prove that $P_2$ cannot be too short. According to the assumptions of the lemma, $|P_2| = 0$ is impossible, because then $u = v$, and we have a cycle in $T$, since $P_1 \subseteq T$. Also, $|P_2| = 1$ is impossible, because then $uv \in G$, $d_T(u,v) \geq t + 1$, and $T$ is a t-spanner of $G$. So, $P_2$ contains at least one node other than $u,v$.

Now, assume that for every node $w \in P_2$: $d_T(M,w) > \lfloor \frac{t-1}{2} \rfloor$. We use this assumption to find another cycle $C' \subseteq G$, composed of two paths $P_1'(u',v')$ and $P_2'(u',v')$. These two paths will preserve the assumptions of the lemma:

1. $V(P_1') \cap V(P_2') = \{u',v'\}$,

¹$H$ stands for hypothesis, and $C$ stands for conclusion.
2. \( |P'_1| > t \),

3. \( P'_1 \subseteq T \), and

4. \( M \) is a \( t \)-midst of \( P'_1 \).

Moreover, we will see that \( P'_2 \subseteq P_2 \). This implies two things. First, the assumption \( \neg C(M, P_2) \) implies that \( \neg C(M, P'_2) \). Second, \( |P'_2| < |P_2| \). This process continues. Eventually, since at each iteration the new \( P_2 \) is strictly shorter than the previous one, we will have a cycle \( C^* \), composed of two paths \( P'_1 \), and \( P'_2 \), such that \( |P'_2| \leq 1 \), which contradicts \( |P_2| \geq 2 \). The description, and the proof of correctness of the construction follows.

Construction: Define as \( Q_u \) the subpath of \( T \) between \( M \) and \( u \), and as \( Q_v \), the subpath of \( T \) between \( M \) and \( v \). We note that \( E(Q_u) \cap E(Q_v) = 0 \). Moreover, \( |Q_u| > \left\lfloor \frac{1}{t} \right\rfloor \), and \( |Q_v| > \left\lfloor \frac{1}{t} \right\rfloor \), since \( M \) is a midst of \( P \).

Since \( |P_2| \geq 2 \), we consider a node \( w \in V(P_2) \setminus \{u, v\} \) such that, if \( Q' \subseteq T \) is the unique shortest path between \( M \) and \( w \), then \( Q' \cap (P_2 \setminus \{u, v\}) = \{w\} \). We can say that \( w \) is a first node of \( P_2 \setminus \{u, v\} \) we meet while walking on \( T \), towards \( P_2 \setminus \{u, v\} \), having started from \( M \). Since \( T \) is connected there is always such a path.

Here, \( Q' \) will provide us with the definition of the new cycle \( C' \). Also, \( Q' \) can share nodes, other than \( V(M) \), with only one of \( Q_u, Q_v \). More precisely, if \( A = (V(Q') \cap V(Q_u)) \setminus V(M) \neq 0 \), and \( B = (V(Q') \cap V(Q_v)) \setminus V(M) \neq 0 \), then \( T \) contains a cycle. Thus, we have to consider only three cases:

- \( A \neq \emptyset \) and \( B = \emptyset \). Then set \( u' = w \), and \( v' = v \), and consider \( C' \) to be the union of \( P'_1 = Q' \cup Q_v \cup M \), and \( P'_2 \), the path on \( P_2 \) connecting \( w \) and \( v \). See figure 3.2.

- \( B \neq \emptyset \) and \( A = \emptyset \). Then set \( u' = u \), and \( v' = w \), and consider \( C' \) to be the union of \( P'_1 = Q' \cup Q_u \cup M \), and \( P'_2 \), the path on \( P_2 \) connecting \( w \) and \( u \).

- \( A = B = \emptyset \). If \( |M| = 0 \) (\( t \) is odd), use for \( C' \) either of the above two definitions.

In the case that \( |M| = 1 \), we define \( C' \) as follows. If \( V(Q') \cap V(Q_u) \neq \emptyset \), then take the definition for \( C' \) from the first case, otherwise \( (V(Q') \cap V(Q_v) \neq \emptyset) \) from the second case. Note that, when \( |M| = 1 \), we cannot have both \( V(Q') \cap V(Q_u) \neq \emptyset \), and \( V(Q') \cap V(Q_v) \neq \emptyset \), since \( Q' \), as a shortest path, cannot contain both nodes of \( M \).

Now, we prove that \( P'_1 \) and \( P'_2 \) preserve assumptions 1-4 of the lemma. In all the above cases, \( V(P'_1) \cap V(P'_2) = \{u', v'\} \), since \( Q' \cap (P_2 \setminus \{u, v\}) = \{w\} \). According to our
assumption that $\neg C(M, P_2)$ holds, we know that $d_T(M, w) > \lfloor \frac{t-1}{2} \rfloor$; so, $|Q'| \geq \lfloor \frac{t-1}{2} \rfloor + 1$. Thus, $|P'_1| \geq \lfloor \frac{t-1}{2} \rfloor + 1 + \lfloor \frac{t-1}{2} \rfloor + 1 + |M| = t + 1 > t$. Of course, $P'_1 \subseteq T$, since $Q' \subseteq T$. Finally, $M \subseteq P'_1$, and $M$ is a $t$-midst of $P'_1$, again because of the assumption that $d_T(M, w) > \lfloor \frac{t-1}{2} \rfloor$. Hence, $P'_1$ and $P'_2$ preserve assumptions 1-4 of the lemma, i.e. $H(P'_1, P'_2)$.

Furthermore, $P'_2 \subseteq P_2$, since $w$ is on $P_2$, and $w \notin \{u, v\}$. So, $\neg C(M, P'_2)$ holds, as well. Moreover, $|P'_2| < |P_2|$.

Now, if $|P'_2| \leq 1$, we reach a contradiction, according to the second paragraph of the proof. If $|P'_2| > 1$, we can reapply the same method to construct another cycle $C''$, where $|P''_2| < |P'_2|$. Applying this method iteratively, eventually we reach a contradiction. □

Note that this lemma was first used to support a proof of theorem 6. That reflects the way it is stated, and its proof follows terminology related to cycles in tree $t$-spanner admissible graphs. Another way to consider this lemma is illustrated in figure 3.3. Let $T$ be a tree $t$-spanner of a graph $G$, where $t$ is odd. Here, $M$ is a midst of a long enough tree path, and $W$ is the set of nodes of the graph, such that their tree distance from $M$ is less than or equal to $\lfloor \frac{t-1}{2} \rfloor$, exactly as they are described in the lemma. The dashed lines in this figure mark components of $T \setminus M$. Now, the lemma says in other words that for every pair of nodes $u, v$ in $G \setminus W$ if they belong to different components of $T \setminus M$, then they belong to different components of $G \setminus W$. So, there cannot be any edge such as $u^*u^*$ as shown in figure 3.3. Actually, such an edge $u^*u^*$ is constructed in the proof of lemma 2, in order to reach a contradiction. Moreover, the description of this situation leads to a slightly different proof of this lemma, since it is not necessary to involve any iterative construction.

Figure 3.2: Construction of the new paths $P'_1$, and $P'_2$. 
Last but not least, the importance of set $W$ being a cutset of graph $G$, leads naturally to the definition of $t$-star graphs, which are introduced in the next section.

### 3.3 A decomposition theorem on tree spanners

In order to state the theorem of this section, we introduce the concept of a $t$-star, which is actually a generalization of the well known star. Recall that a star is a $K_{1,k}$, for some positive number $k$. We discuss possible $t$-star cuts of a tree $t$-spanner admissible graph. The existence of such a cut in a graph provides a decomposition of the graph.

#### 3.3.1 The $t$-star graph

We start with the definition of a $t$-star. Afterwards, we explain a few characteristics of this family of graphs, and present some examples. The definition of a $t$-star is very close to the
definition of a $t$-midst of a path.

**Definition 3** Let $t \geq 0$. A $t$-star is a graph $G$ which is either a $(t-1)$-star (when $t > 0$), or:

- if $t$ is even, there exists at least one node $x$ in $G$, such that $\forall u \in G$, $d(x, u) \leq \lfloor \frac{t}{2} \rfloor$. A center $K(G, t)$ of $t$-star $G$ is a one node subgraph of $G$ consisting of such an $x$.

- if $t$ is odd, there exists at least one edge $xy$ in $G$, such that $\forall u \in G$, $d(\{x, y\}, u) \leq \lfloor \frac{t}{2} \rfloor$. A center $K(G, t)$ of $t$-star $G$ is a two node subgraph of $G$ consisting of such an edge $xy$.

As the midst of a path, the center $K$ of a $t$-star is a path too, and $|K| = |E(K)|$. For example a 0-star is only one node, a 1-star is just an edge or a node, a 2-star is a graph that contains a spanning star or is a 1-star, and a 3-star is a graph that contains a dominating edge or is a 2-star. For an illustration of small $t$-stars see figure 3.4.

![Figure 3.4: Examples of small t-stars](image)

If $k \leq l$ then a $k$-star is also an $l$-star. Moreover a $t$-star is a connected graph, and every connected graph is a $t$-star, for some $t$. For example, a path of $d$ edges is a $d$-star, and has a unique center.

Furthermore, we say that a connected graph $G$ has a $t$-star cut $S$, if $S$ is a $t$-star, and $G \setminus S$ is a disconnected graph. The above definition and lemma 2 are the basis of the condition we present in this section.

There are some differences between a $t$-star and a graph which has diameter at most $t$. Although a $t$-star always has diameter less than or equal to $t$, the inverse does not hold. For example, a cycle with 8 nodes has diameter 4, but it is not a 4-star. The following lemma is on the diameter of a spanning tree of a graph.

**Lemma 3** Consider a connected graph $G$, which is not a $t$-star for some fixed $t$. Let $T$ be a spanning tree of $G$. Then, the diameter $d$ of $T$ is strictly greater than $t$. 

29
Proof. We prove this lemma by contradiction. So, assume that \( d \leq t \). Consider a longest path \( P(u, v) \) of \( T \), then \( d = |E(P)| \). So, \( P \) is a \( d \)-star. Consider the center \( K(P, d) \) of \( P \). We know that \( d_T(K, u) \leq \lfloor \frac{d}{2} \rfloor \), and \( d_T(K, v) \leq \lceil \frac{d}{2} \rceil \). Moreover, for every node \( p \in G \) (\( T \) is a spanning subgraph of \( G \)), \( d_T(K, p) \leq \lfloor \frac{d}{2} \rfloor \), because otherwise \( P \) would not be a longest path. But, for every node \( p \in G \), \( d_G(K, p) \leq d_T(K, p) \), since by adding edges we cannot increase the distance between two nodes. Also, \( \lfloor \frac{d}{2} \rfloor \leq \lceil \frac{d}{2} \rceil \), since \( d \leq t \). So \( G \) is a t-star, which is a contradiction.

Furthermore, given a graph \( G \) and a set of vertices \( W \) in \( G \), we can find in polynomial time the shortest paths from \( W \) to all the other nodes in \( G \), using Dijkstra's algorithm, thus providing an easy polynomial algorithm to determine if a graph is a t-star.

**Proposition 3** A t-star graph can be recognized in polynomial time.

The basic steps of such an algorithm are the following. Assume that we are given a graph \( G \), and a positive integer \( t \). If \( t \) is even, then for every vertex \( v \) in \( G \) find the distances from \( v \) to the rest of the vertices in \( G \). If during this process we find a vertex \( v \), such that all such distances are less than or equal to \( t/2 \), then \( G \) is a t-star, otherwise \( G \) is not a t-star. In the case that \( t \) is odd, we go through the same process for every edge of \( G \).

### 3.3.2 Tree t-spanner vs t-star

In this section, we present a necessary and sufficient condition for a graph to be tree spanner admissible. First, we present a simple lemma.

**Lemma 4** Consider two trees \( T_1 \) and \( T_2 \). If \( T_1 \cap T_2 \) is a tree then \( T_1 \cup T_2 \) is a tree.

**Proof.** Let \( T = T_1 \cap T_2 \) be a tree. First, since \( T \neq \emptyset \), then \( F = T_1 \cup T_2 \) is connected. Assume that \( F \) contains a cycle \( C \). There are some edges of \( C \), which are not in \( T \), because otherwise \( T \) has a cycle. Consider an edge \( e \in C \), such that \( e \in T_1 \) (w.l.o.g.) but \( e \notin T \). Consider the longest path \( P \) on \( C \), such that \( e \in E(P), E(P) \subseteq E(C) \cap E(T_1) \), and \( E(P) \cap E(T) = \emptyset \). Consider two cases:

- If \( E(P) = E(C) \), then \( P \) is a cycle which is in \( T_1 \), a contradiction.
- If \( E(P) \neq E(C) \), then \( P \) connects two nodes \( a \) and \( b \). Here, \( P \) is the longest such path, so the neighboring edges of \( P \) on \( C \), belong to \( T_2 \). This implies that both nodes
a, b belong to T. Since T is connected, there is a path \( Q(a, b) \subset T \subset T_1 \). Moreover, 
\( P(a, b) \subset T_1 \), and \( P(a, b) \not\subset T \). So, \( T_1 \) has a cycle, and we again reach a contradiction.

So, \( F \) is connected, and does not contain any cycle. Hence \( F \) is a tree. \( \square \)

As an aside we note that the converse of Lemma 4 also holds. Actually, if \( T_1 \cap T_2 \) is not connected, then there is a path of \( T_1 \) connecting two components, and another path of \( T_2 \) connecting these two components. These two paths form a cycle in \( T_1 \cup T_2 \), a contradiction.

Finally, we are in a position to state and prove the following decomposition characterization of tree \( t \)-spanner admissible graphs. To the best of our knowledge this theorem, our second major contribution, is the first nontrivial characterization of tree \( t \)-spanner \( (t > 2) \) admissible graphs.

**Theorem 7** A graph \( G \) is tree \( t \)-spanner admissible if and only if either

1. \( G \) is a \( t \)-star, or

2. There exist graphs \( L, R \) with the following properties:

   (a) \( G = L \cup R \),

   (b) \( L \cap R \) is a \( (t - 1) \)-star,

   (c) \( L \) has a tree \( t \)-spanner \( T_L \),

   (d) \( R \) has a tree \( t \)-spanner \( T_R \), and

   (e) \( T_L \cap T_R \) is a tree \( t \)-spanner of \( L \cap R \).

**Proof.** (Sufficiency) We prove that \( G \) is tree \( t \)-spanner admissible if either of the above two cases holds.

If \( G \) is a \( t \)-star, let \( K \) be the center of that \( t \)-star. Running Dijkstra's algorithm starting from \( K \), we have a spanning tree \( T \) of \( G \) such that \( d_T(K, v) = d_G(K, v) \leq \lfloor \frac{t}{2} \rfloor \), for all \( v \) in \( G \). Now, for every pair of nodes \( u, v \in G \):

\[
d_T(u, v) \leq d_T(K, v) + d_T(K, u) + |K| \leq \lfloor \frac{t}{2} \rfloor + \lfloor \frac{t}{2} \rfloor + |K| = t,
\]

since \( |K| = t \mod 2 \). Thus, \( T \) is a \( t \)-spanner of \( G \).

For the latter case, \( G \) is the union of two graphs \( L \), and \( R \), which preserve properties \( (a) - (e) \). We prove that \( T = T_L \cup T_R \) is a tree \( t \)-spanner of \( G \).
By property (e), \(T_L \cap T_R\) is a tree. Also, by property (b), \(L \cap R \neq \emptyset\), so \(T_L \cap T_R \neq \emptyset\). Hence, from lemma 4, \(T\) is a tree. We prove that \(T\) is a \(t\)-spanner as well. According to proposition 1, it suffices to show that for every edge \(xy \in G\), \(d_T(x, y) \leq t\). Since \(T_L\) and \(T_R\) are tree spanners, they are spanning trees of \(L\) and \(R\) respectively. So, \(V(T_L \cup T_R) = V(L \cup R) = V(G)\). Moreover, by property (a), \(E(G) = E(L) \cup E(R)\); hence, there are only two possible cases for every edge \(xy \in G\):

- \(xy \in L\). Then \(d_T(x, y) = d_{T_L}(x, y) \leq t\), since \(T_L\) is a tree \(t\)-spanner of \(L\).
- \(xy \in R\). Then \(d_T(x, y) = d_{T_R}(x, y) \leq t\), since \(T_R\) is a tree \(t\)-spanner of \(R\).

So, \(T\) is a spanning tree of \(G\), and \(T\) approximates the distances with factor \(t\). Hence, \(G\) is tree \(t\)-spanner admissible. Note that \(\{xy : x \in L \setminus R, y \in R \setminus L\} \cap E(G) = \emptyset\), so \(L \cap R\) is a cut set of \(G\).

(Necessity) We will prove the equivalent proposition: If \(G\) is not a \(t\)-star, and \(G\) has a tree \(t\)-spanner, then case 2 of the theorem holds for \(G\).

So, consider that \(G\) is not a \(t\)-star and let \(T\) be a tree \(t\)-spanner of \(G\). Since \(T\) is a spanning tree, from lemma 3, the diameter \(d\) of \(T\) is strictly greater than \(t\); so, \(d \geq t + 1\). Thus, there is a path \(P_1(u, v) \subseteq T\), such that \(|P_1| = t + 1\). Consider now the \(t\)-midst \(M(P_1, t)\) of \(P_1\) (\(P_1\) has only one \(t\)-midst). We see that \(|M| = (t + 1) \mod 2 = (t - 1) \mod 2\), so the center \(K\) of a \((t - 1)\)-star has the same size as \(M\).

Let \(S\) be the subgraph of \(G\) having as vertex set \(V(S) = \{p \in V(G) : d_T(M, p) \leq \lceil \frac{t - 1}{2} \rceil\}\), and edge set \(E(S) = \{ab \in E(G) : a, b \in V(S)\}\). Since \(\forall a, b \in S\) \(d_S(a, b) \leq d_T(a, b)\), \(S\) is a \((t - 1)\)-star, and \(M(P_1, t)\) is its center \(K(S, t)\).

Fact 1: \(G \setminus S\) is disconnected.

We know that \(u, v \notin S\), since \(M\) is a \(t\)-midst of \(P_1(u, v)\). So, \(\{u, v\} \subseteq G \setminus S\). We prove by contradiction that there is not a path between \(u\) and \(v\) in \(G \setminus S\).

So, assume that there is a path \(P_2(u, v) \subseteq G \setminus S\). From the definition of \(S\), \(V(P_1) \setminus V(S) = \{u, v\}\), so \(V(P_1) \cap V(P_2) = \{u, v\}\). Moreover \(|P_1| \geq t + 1\), and \(P_1 \subseteq T\). Also \(T\) is a tree \(t\)-spanner of \(G\). So, \(P_1\) and \(P_2\) compose a cycle \(C\) having the properties of lemma 2. From the definition of \(P_2\), for every node \(w \in P_2\) we know that \(w \notin V(S) = \{p \in V(G) : d_T(M, p) \leq \lceil \frac{t - 1}{2} \rceil\}\). So we have a contradiction, according to the same lemma. Hence, there is not a path between \(u\) and \(v\) in \(G \setminus S\). So, \(S\) is a \((t - 1)\)-star cut of \(G\), and fact 1 holds.
Now, $G \setminus S$ is not connected. There is one component of $G \setminus S$, which contains node $u$. Call this component $H_1$. Define as $H_2$ the set of the remaining components, or more formally, $H_2 = (G \setminus S) \setminus H_1$. Using the definitions of $S$, $H_1$, and $H_2$, we define graphs $L$ and $R$. Let $V(L) = V(S) \cup V(H_1)$ and $E(L) = E(G) \cap V(L)^2$; let $V(R) = V(S) \cup V(H_2)$ and $E(R) = E(G) \cap V(R)^2$.

Fact 2: $L$ and $R$ satisfy properties (a) – (e).

For property (a), $V(L) \cup V(R) = V(H_2) \cup V(S) \cup V(H_1) = ((V(G) \setminus V(S)) \setminus V(H_1)) \cup V(S) \cup V(H_1) = V(G)$. Moreover, $E(L) \cup E(R) = (E(G) \cap V(L)^2) \cup (E(G) \cap V(R)^2) = E(G) \cap V(L)^2 \cup V(R)^2 = E(G) \cap ((V(L) \cup V(R))^2 \setminus ((V(L) \setminus V(R)) \times (V(R) \setminus V(L))))$. But $E(G) \cap ((V(L) \setminus V(R)) \times (V(R) \setminus V(L))) = \emptyset$ since $S$ is a cut set and $(V(L) \cup V(R))^2 = V(G)^2$. Therefore $E(L) \cup E(R) = E(G)$. Hence $G = L \cup R$.

We see that $V(H_1) \cap V(H_2) = \emptyset$, so $V(L) \cap V(R) = V(S)$. Also $E(L) \cap E(R) = E(G) \cap (V(L)^2 \cap V(R)^2) = E(G) \cap V(S)^2 = E(S)$. So $L \cap R = S$ and $S$ is a $(t - 1)$-star. Hence, property (b) holds.

Here $G$ has a tree $t$-spanner $T$. Set $T_L = L \cap T$. Also set $T_R = R \cap T$. From definition of $S$, we see that $S \cap T$ is a tree $t$-spanner of $S$. Moreover $T_L \cap T_R = (L \cap R) \cap T = S \cap T$. Therefore $T_L \cap T_R$ is a tree $t$-spanner of $L \cap R$; so property (e) holds.

Since $E(L) \subset E(G)$, $E(R) \subset E(G)$ and $T$ is a tree $t$-spanner of $G$, it suffices to prove that $T_L$ and $T_R$ are connected in order for properties (c), and (d) to hold. Assume that $T_L$ is not connected. So, there are at least two components in $T_L$. Since $T$ is connected, there is a path $P_T \in T$ between these two components and $P_T \cap L = \emptyset$. Since there are no edges of $G$ between $L \setminus S$ and $G \setminus L$, $P_T$ connects two nodes of $S$. But $T \cap S$ is connected, so $T$ contains a cycle, which is a contradiction. Hence $T_L$ is a tree $t$-spanner of $L$. Using exactly the same argument (replace $L$ with $R$), we prove that $T_R$ is connected as well. This completes the proof of fact 2.

We have proved that there exist graphs $L$ and $R$, which satisfy properties (a) – (e). This completes the proof, according to the first paragraph of the proof of necessity. □

3.4 Relation to previous work

We now show that theorem 7 subsumes various existing results. This theorem refers to all positive values of $t$. As we have mentioned before, a graph is tree 1-spanner admissible if
and only if it is a tree. So, when \( t = 1 \), this theorem describes a necessary and sufficient condition for a graph to be a tree in terms of decomposition. Indeed, if a tree is not a one vertex graph, or a one edge graph, then it has a one vertex cut, which is a 0-star cut, and the resulting components are trees. For the sufficient part, we can compose a bigger tree by taking two smaller trees, and making them coincide only on one vertex.

It is interesting to note that theorem 4, which is a characterization of tree 2-spanner admissible graphs, is an immediate corollary of theorem 7, when \( t = 2 \). To see this, note the following for a tree 2-spanner admissible graph \( G \). First, if \( G \) has a universal vertex, then it is a 2-star. Second, an articulation point is a 0-star cut, which is a (2-1)-star cut as well, so theorem 7 partitions \( G \) into blocks, which recursively are tree 2-spanner admissible graphs. Third, the last condition of theorem 4 refers to a vertex cut of \( G \) that is a pair of adjacent vertices, which is a 1-star cut. Moreover, this pair of vertices has to remain connected in a tree 2-spanner of \( G \), a condition which is included in part 2 of theorem 7.

Furthermore, as noted in chapter 1, Leizhen Cai has presented a result on tree 3-spanner admissible graphs (theorem 5). This theorem can be viewed as a necessary condition, if we write it in the following form: If \( G \) is a tree 3-spanner admissible graph, then \( G \) contains either a dominating edge, or a star cut. Theorem 5 also implies that a graph that contains a dominating edge is a tree 3-spanner admissible graph. Since a star is a 2-star, and a graph that contains a dominating edge is a 3-star, theorem 5 is an immediate corollary of theorem 7.
Chapter 4

Conclusions and future work.

In this chapter we present briefly the results of this thesis, and we propose how these results can be used in the future. Moreover, we discuss the algorithmic aspects of theorem 7.

4.1 Conclusions.

Theorem 6 provides a condition for all the cycles of a tree \( t \)-spanner admissible graph. This condition does not hold for spanning trees. So, this theorem gives another explanation of why the tree \( t \)-spanner problem is more difficult than the well known minimum spanning tree problem. Moreover, corollary 1 illustrates a structural difference between a tree \( 2 \)-spanner admissible graph, and a graph that has stretch factor greater than 2. Recall that the complexity status of the tree 3-spanner problem is unknown.

The study of the behavior of a cycle in a tree \( t \)-spanner admissible graph led to lemma 2, which gave rise to the definition of \( t \)-star graphs. Since such a graph can be a cut set of a tree \( t \)-spanner admissible graph, the generalization of theorems 4 and 5 became clear. Such a generalization is provided by theorem 7, which is a characterization of tree \( t \)-spanner admissible graphs, in terms of decomposition.

4.2 The algorithmic aspects of theorem 7.

Theorem 7 states that if a graph is a \( t \)-star, then it is tree \( t \)-spanner admissible. We can easily check in polynomial time if a graph \( G \) is a \( t \)-star, according to proposition 3.

But, there can be graphs that are not \( t \)-stars, although they are tree \( t \)-spanner admissible.
Such graphs contain a \((t - 1)\)-star cut, according to theorem 7. The proof of this theorem is existential, so this theorem does not provide us with an efficient algorithm. Of course we expected this, since the tree \(t\)-spanner problem is shown to be NP-complete for \(t \geq 4\) (theorem 3). Moreover, even for \(t = 3\) we cannot easily achieve an efficient algorithm. Indeed, in the worst case there can be exponentially (in the size of the graph) many \((3 - 1)\)-star cuts, so we cannot go through all these cases efficiently. Note that a 1-star cut is at most a pair of vertices, so there cannot be too many 1-star cuts. For the tree 2-spanner problem we are interested in 1-star cuts, hence there is a linear time algorithm for this problem \([9]\). Another comment on theorem 7 is that in the proof of sufficiency we do not use all the power of the condition that the cut set is a \((t - 1)\)-star. In order to resolve this asymmetry a better, and more algorithmic statement of this theorem is required.

Despite these difficulties, we strongly believe that theorem 7 can help us to construct an efficient algorithm that solves the tree \(t\)-spanner \((t \geq 4)\) problem for graphs with maximum degree bounded by a constant. Indeed, in this case the number of possible \((t - 1)\)-star cuts in such a graph is polynomial in the size of the graph. Moreover, we believe that for \(t = 3\) there will be an efficient algorithm for general graphs. This intuition is supported by the fact that the same situation holds for planar graphs \([16]\), as mentioned in chapter 1.

Furthermore, theorem 7 may give rise to an approximation algorithm for the tree \(t\)-spanner problem for general graphs. Actually, the basic problem for an efficient algorithm is that there can be exponentially many star cuts, and only some of them can lead to a tree \(t\)-spanner. Now, assume that we are given a tree \(t\)-spanner admissible graph \(G\). For all such graphs, we believe that we can efficiently determine a good enough decomposition of \(G\), so that we can always get a tree \((\alpha \cdot t)\)-spanner of \(G\) for some constant \(\alpha > 1\).

### 4.3 Future work

Besides the future work from the algorithmic point of view that we described in the previous section, some more open problems are motivated by this thesis. Theorem 6 is a necessary condition for a cycle in a tree \(t\)-spanner admissible graph, but does not refer to the structure of the graph around this cycle. Since this theorem applies to all the cycles of such a graph, we can apply it simultaneously to many cycles, and get better results. For example this approach led to corollary 1. Through another approach we can examine what happens when
this cycle is a maximal one. Moreover, we believe that there is an interesting lower bound on the number of leaves of a tree $t$-spanner of a graph.

The two theorems of this thesis seem to be independent to each other. Since they refer to the same problem, there might be a connection between them. Theorem 6 involves graph theoretical aspects of the tree $t$-spanner admissible graphs, while theorem 7 obviously has algorithmic applications as well. Is there a theorem that captures the ideas of both of them?
Bibliography


[22] C. Levcopoulos and A. Lingas. There are planar graphs almost as good as the complete graphs and almost as cheap as minimum spanning trees. Algorithmica, 8:251–256, 1992.


Appendix A

Appendix

The following definitions are not given in the body of the thesis. The terms are presented in alphabetical order. For some of the definitions we used [6].

A chordless cycle in a graph $G$ is an induced subgraph of $G$ isomorphic to a $C_t$ for some $t \geq 4$. A graph $G$ is chordal if it has no chordless cycle. A $p$-sun $S_p$ ($p \geq 3$) is a chordal graph on $2p$ vertices whose vertex set can be partitioned into two sets $U = \{u_1, \ldots, u_p\}$ and $W = \{w_1, \ldots, w_p\}$, such that $W$ is independent and every $w_i$ is adjacent only to $u_i$ and $u_{i+1}$ (mod $p$). A chordal graph $G$ is called strongly chordal if it does not contain any $p$-sun as an induced subgraph for some $p$.

A cograph (complement reducible) is either a single vertex or the complement of a cograph or the join of two cographs.

Let $G$ be a bipartite graph with partite sets $X$ and $Y$. An ordering of $X$ in $G$ has the adjacency property if for each vertex $y \in Y$, its neighborhood consists of vertices that are consecutive in this ordering. A bipartite graph is convex if there is an ordering of at least one of the two partite sets which fulfills the adjacency property.

A $k$-cube graph consists of the vertices and edges of a $k$-dimensional cube.

A cycle double cover of a graph $G$ is a collection of cycles of $G$ such that each edge of $G$ belongs to exactly two cycles of this collection.

A graph $G$ is an interval graph if and only if there exists a set of intervals on the real line such that for every vertex $v \in V(G)$ there exists a corresponding interval $I_v$ in this set such that $uv \in E(G)$ if and only if $I_u \cap I_v \neq \emptyset$.

A graph $G$ is a permutation graph if and only if there exists a set of lines in the plane
such that for every vertex \( v \in V(G) \) there exists a corresponding line \( L_v \) in this set such that \( uv \in E(G) \) if and only if \( L_u \cap L_v \cap T \neq \emptyset \), where \( T \) is the region between two parallel lines, which intersect all the lines in this set.

A **planar graph** is a graph that can be drawn in the plane without edge crossings. A **plane graph** is a particular drawing of a planar graph in the plane with no crossings. The **faces** of a plane graph are the maximal regions of the plane that are disjoint from the drawing. The **length** of a face in a plane graph \( G \) is the length of the walk in \( G \) that bounds it. The **dual graph** \( G^* \) of a plane graph \( G \) is a plane graph having a vertex for each region in \( G \). The edges of \( G^* \) correspond to the edges of \( G \) as follows: if \( e \) is an edge of \( G \) that has region \( X \) on one side and region \( Y \) on the other side, then the corresponding dual edge \( e^* \) is an edge joining the vertices \( x, y \) of \( G^* \) that correspond to the faces \( X, Y \) of \( G \).

A simple graph \( G \) is a **split graph** if \( V(G) \) can be partitioned into \( Q \) and \( S \) such that \( Q \) induces a clique and \( S \) is an independent set.

A vertex in a graph \( G \) is **\( k \)-simplicial** if its neighborhood in the graph induces a clique of \( k \) vertices. A graph \( T_k \) is a **\( k \)-tree** if it is either a complete graph of \( k \) vertices or it has a \( k \)-simplicial vertex \( x \) such that \( T_k \setminus \{x\} \) is a \( k \)-tree.

A **tritree** of a graph \( G \) is a spanning tree \( T \) of \( G \) such that every fundamental cycle of \( G \) with respect to \( T \) is a triangle.