A Tourist Guide through Treewidth

Hans L. Bodlaender*
Department of Computer Science, Utrecht University
P.O.Box 80.089, 3508 TB Utrecht, the Netherlands

Abstract

A short overview is given of many recent results in algorithmic graph theory that deal with the notions treewidth, and pathwidth. We discuss algorithms that find tree-decompositions, algorithms that use tree-decompositions to solve hard problems efficiently, graph minor theory, and some applications. The paper contains an extensive bibliography.

1 Introduction

In recent years, the notions 'treewidth', 'pathwidth', 'tree-decomposition', and 'path-decomposition' have received a growing interest. These notions underly several important and sometimes very deep results in graph theory and graph algorithms, and are very useful for the analysis of several practical problems.

In this paper, we give an overview of a number of these applications, and algorithmic results. In section 2 we give the main definitions. Applications of the notions discussed in this paper are given in section 3. In section 4 we explain the basic idea behind linear time algorithms on graphs with constant bounded treewidth. In section 5 we review some results that deal with graph minors. In section 6 we discuss algorithms that find 'suitable' tree- or path-decompositions.

It should be noted that the constant factors, hidden in the 'O'-notation can be quite large for several of the algorithms, discussed in this paper. In many cases, additional ideas will be required to turn the methods, described here, into really practical algorithms.

2 Definitions

In this section we give the most important definitions, with an example. The notions of treewidth and pathwidth were introduced by Robertson and Seymour [109, 115].

*email: hansb@cs.ruu.nl. This work was partially supported by the ESPRIT Basic Research Actions of the EC under contract 7141 (project ALCOM II).

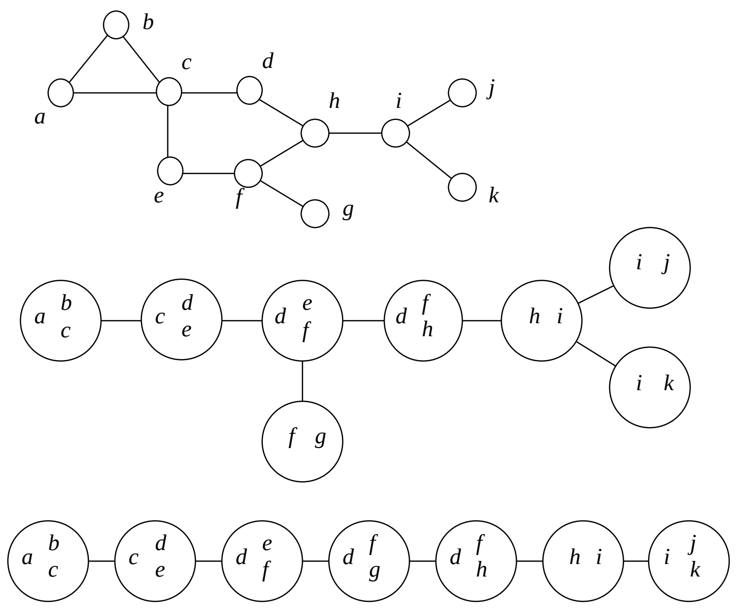

Figure 1: Example of a graph with tree- and path-decomposition

Definition. A tree-decomposition of a graph $G = (V, E)$ is a pair $(\{X_i \mid i \in I\}, T = (I, F))$ with $\{X_i \mid i \in I\}$ a family of subsets of V, one for each node of T, and T a tree such that

- $\bigcup_{i \in I} X_i = V$.
- for all edges $(v, w) \in E$, there exists an $i \in I$ with $v \in X_i$ and $w \in X_i$.
- for all $i, j, k \in I$: if j is on the path from i to k in T, then $X_i \cap X_k \subseteq X_j$.

The treewidth of a tree-decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ is $\max_{i \in I} |X_i| - 1$. The treewidth of a graph G is the minimum treewidth over all possible tree-decompositions of G.

The notion of pathwidth is defined similarly. Now T must be a path.

Definition. A path-decomposition of a graph $G = (V, E)$ is a sequence of subsets of vertices $(X_1, X_2, \ldots, X_r)$, such that

- $\bigcup_{1 \leq i \leq r} X_i = V$.
- for all edges $(v, w) \in E$, there exists an i, $1 \leq i \leq r$, with $v \in X_i$ and $w \in X_i$.
- for all $i, j, k \in I$: if $i \leq j \leq k$, then $X_i \cap X_k \subseteq X_j$.

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8
N_1	1	1	0	0	0	1	1	0
N_2	0	0	0	1	1	0	0	0
N_3	0	0	1	0	0	0	1	0
N_4	0	1	0	1	0	0	0	1
N_5	0	0	1	0	1	1	0	0

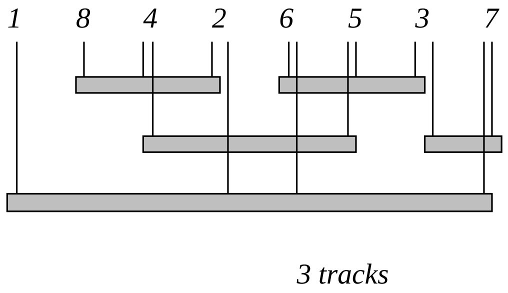

Figure 2: Example of gate matrix layout

The pathwidth of a path-decomposition $(X_1, X_2, \ldots, X_r)$ is $\max_{1 \leq i \leq r} |X_i| - 1$. The pathwidth of a graph G is the minimum pathwidth over all possible path-decompositions of G.

In figure 1, an example of a graph with treewidth and pathwidth 2 is given, together with a tree- and path-decomposition of it.

Clearly, the pathwidth of a graph is at least its treewidth. There are several equivalent characterizations of the notions of treewidth and pathwidth, see e.g. [3, 15, 18, 99, 143]. The (probably) most well known equivalent characterization of treewidth is by the notion 'partial k-tree', see [132, 139]. Also, tree decompositions are reflected by graph expressions, where graphs are built by operations on graphs with some special vertices (the *sources*) like: parallel composition, forget sources, renaming of sources. The treewidth can be characterized in terms of the number of sources used in the operations. See [50].

3 Applications

Several well-studied graph classes have bounded treewidth or pathwidth, hence many results discussed here also apply for these classes. Examples are trees (treewidth 1), series-parallel graphs (treewidth 2), outerplanar graphs (treewidth 2), and Halin graphs (treewidth 3). See e.g. [18, 20, 132, 143]. We mention some other applications.

3.1 VLSI layouts

A well studied problem in VLSI layout theory is the Gate Matrix Layout problem. This problem is stated in terms of a matrix $M = (m_{ij})$, whose columns represent gates $G_1, \ldots, G_n$, and whose rows represent nets $N_1, \ldots, N_m$. If $m_{ij} = 1$, then net N_i must be connected with gate G_j. An example is given in figure 2. The problem of finding a permutation of the gates, such that all nets can be made within the minimum number of tracks is equivalent to the pathwidth problem (see [63]). See [99] for an extensive overview. See also [53].

3.2 Cholesky factorization

There is also a close connection between treewidth, and Choleski factorization on sparse symmetric matrices.

In the multifrontal method for Choleski factorization, one step is of the form

$$\begin{bmatrix} d & v^T \\ v & B \end{bmatrix} = \begin{bmatrix} \sqrt{d} & 0 \\ v/\sqrt{d} & I \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & B - v \cdot v^T/d \end{bmatrix} \cdot \begin{bmatrix} \sqrt{d} & v^T/\sqrt{d} \\ 0 & I \end{bmatrix}$$

where v is an $(n-1)$-vector, and B is an $n-1$ by $n-1$ maxtrix. I is the $n-1$ by $n-1$ identity matrix. The process is repeated with the matrix $B - v \cdot v^T$. Consider the graph with vertices $1, 2, \ldots, n$, and edges between vertices i and j, if the matrix entries on positions (i,j) and (j,i) are non-zero. One step as described above corresponds to removing a vertex and connecting all its neighbors. As the matrix is sparse, one wants to find an order of colums/rows to be eliminated for which all matrices $v \cdot v^T$ are small, i.e. have a large number of columns and rows that are entirely 0. One can show that to bound the maximum size of these matrices corresponds to bounding the treewidth of the graph, described above. For more details, see e.g. [29].

3.3 Expert systems

Graphs modelling certain type of expert systems have been observed to have small treewidth in practice. Tree-decompositions of small treewidth for these graphs can be used to perform efficiently certian otherwise time-consuming statistical computations needed for reasoning with uncertainity in these systems. See e.g. [92, 138].

3.4 Evolution theory

Researchers in molecular biology are interested in the problem, given a set of species, a set of characteristics, and for each specie and each characteristic, the value that that characteristic has for that specie, to find a 'good' evolution tree for these species and their possibly extinct ancestors. One variant of this problem is called the PERFECT PHYLOGENY problem. This problem can be shown to be equivalent with the following graph problem: given a graph $G = (V, E)$ with a coloring of the vertices, can we add edges to G such that the resulting graph is chordal but has no edges between vertices of the same color? Equivalently, does there exist a tree-decomposition $(\{X_i \mid i \in I\}, T)$ of G such that for all $i \in I$: if $v, w \in X_i$, $v \neq w$, then v and w have different colors. So, a necessary condition is that the treewidth of G is smaller than the number of colors. See [2, 28, 33, 79, 80, 98].

3.5 Natural language processing

Kornai and Tuza [88] have observed that dependency graphs of sentences encoding the major syntactic relations among the words have usually pathwidth at most 6. The pathwidth closely resembles the *narrowness* of these graphs. For the relationship of this notion to natural language processing, see [88].

4 Bounded treewidth and linear time algorithms

An important reason for the interest in tree-decompositions, is that if we have a tree-decomposition of a graph $G = (V, E)$ with its treewidth bounded by some fixed constant k, then we can solve many problems that are hard (intractable) for arbitrary graphs, in polynomial and often linear time. Problems which can be dealt with in this way include many well-known NP-complete problems, like INDEPENDENT SET, HAMILTONIAN CIRCUIT, STEINER TREE, etc., but also certain statistical computations (including some with applications to reasoning with uncertainity in expert systems [92, 138]), and some PSPACE-complete problems [4, 5, 26]. Results of this type can be found — among others — in [3, 4, 5, 8, 10, 14, 19, 26, 22, 31, 37, 44, 47, 52, 55, 67, 69, 71, 73, 74, 75, 87, 90, 93, 94, 95, 96, 107, 132, 137, 141, 142, 143, 144, 145].

As an example we consider the maximum independent set problem. In this problem, we a looking for the maximum size of a set $W \subseteq V$ in a given graph $G = (V, E)$, such that for all $v, w \in W : (v, w) \notin E$.

Given a tree-decomposition, it is easy to make one with the same treewidth, and with T a rooted binary tree. Suppose we have such a tree-decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ of input graph G, with root of T r, and with treewidth k. For each $i \in I$, define $Y_i = \{v \in X_j \mid j = i \text{ or } j \text{ is a descendant of } i\}$.

Note that if $v \in Y_i$, and $v \in X_j$ for some node $j \in I$ that is not a descendant of i, then by definition of tree-decomposition, $v \in X_i$. Similarly, if $v \in Y_i$, and v is adjacent to a vertex $w \in X_j$ with j a descendant of i, then $v \in X_i$ or $w \in X_i$. As a consequence, we have that, when we have an independent set W of the subgraph induced by Y_i, $G[Y_i]$, and want to extend this to an independent set of G, then important is *only* what vertices in X_i belong to W, not what vertices in $Y_i - X_i$ belong to W. Of the latter, only the *number* of the vertices in W is important.

For $i \in I$, $Z \subseteq X_i$, define $is_i(Z)$ to be the maximum size of an independent set W in $G[Y_i]$ with $W \cap X_i = Z$. Take $is_i(Z) = -\infty$, if no such set exists.

Our algorithm to solve the independent set problem on G basically consists of computing all tables is_i, for all nodes $i \in I$. This is done in a bottom-up manner in the tree: each table is_i is computed after the tables of the children of node i are computed. For a leaf node i, the following formula can be used to compute all $2^{|X_i|}$ values in the table is_i.

$$is_i(Z) = \begin{cases} |Z| & \text{if } \forall v, w \in Z : (v, w) \notin E \\ -\infty & \text{if } \exists v, w \in Z : (v, w) \in E \end{cases}$$

For an internal node i with two children j and k, we have the following formula.

$$is_i(Z) = \begin{cases} \max\{is_j(Z') + is_k(Z'') + |Z \cap (X_i - X_j - X_k)| \\ -|Z \cap X_j \cap X_k| \mid Z \cap X_j = Z' \cap X_i \\ \text{ and } Z \cap X_k = Z'' \cap X_i\} & \text{if } \forall v, w \in Z : (v,w) \notin E \\ -\infty & \text{if } \exists v, w \in Z : (v,w) \in E \end{cases}$$

The idea behind the last formula is: take the maximum over all sets $Z' \subseteq X_j$ that agree with Z in which vertices in $X_i \cap X_j$ belong to the independent set, and similarly for $Z'' \subseteq X_k$. Vertices in $Z \cap X_i - X_j - X_k$ are not counted yet, so their number should be added, while vertices in $Z \cap X_j \cap X_k$ are counted twice, hence their number should be subtracted once.

We compute for each node $i \in I$ the table is_i in some bottom-up order, until we have computed the table is_r. Note that we then can easily find the maximum size of an independent set in G, as this is $\max_{Z \subseteq X_r} is_r(Z)$. Hence, we have an algorithm, that solves the independent set problem on G in $O(2^{3k}n)$ time. (Optimizations can bring the factor 2^{3k} down to 2^k.) It is also possible, by using standard dynamic programming techniques, to construct the maximum sized independent set W itself.

The idea behind this example is: each table entry gives information about an equivalence class of partial solutions. The number of such equivalence classes is bounded by some constant, when the treewidth is bounded by a constant. Tables can be computed using only the tables of the children of the node.

The technique works for many examples. However, there are also results that state that large *classes* of problems can be solved in linear time, when a tree-decomposition with constant bounded treewidth is available. One of the most powerfull results of this type is the result by Courcelle [47, 51, 46], which has been extended by Arnborg et al [8], by Borie et al [38], and by Courcelle and Mosbah [52], on (Extended) Monadic Second Order formulas. These result basically state that each graph problem that is expressible with a formula using the following language constructions: logical operations ($\wedge$, $\vee$, $\neg$, $\Rightarrow$), quantification over vertices, edges, sets of vertices, sets of edges (e.g. $\exists v \in V$, $\forall e \in E$, $\forall W \subseteq V$, $\exists F \subseteq E$), membership tests ($v \in W$, $e \in E$), adjacency tests ($(v,w) \in E$, v is endpoint of e), and certain extensions, can be solved in linear time on graphs with given a tree-decomposition of constant bounded treewidth. The extensions allow not only to deal with decision problems, but also optimization problems (like maximum independent set).

For example, the problem whether a given graph G can be colored with three colors can be stated as

$$\exists W_1 \subseteq V : \exists W_2 \subseteq V : \exists W_3 \subseteq V : \forall v \in V : (v \in W_1 \vee v \in W_2 \vee v \in W_3) \wedge \forall v \in V : \forall w \in W : (v,w) \in E \Rightarrow (\neg(v \in W_1 \wedge w \in W_1) \wedge \neg(v \in W_2 \wedge w \in W_2) \wedge \neg(v \in W_3 \wedge w \in W_3))$$

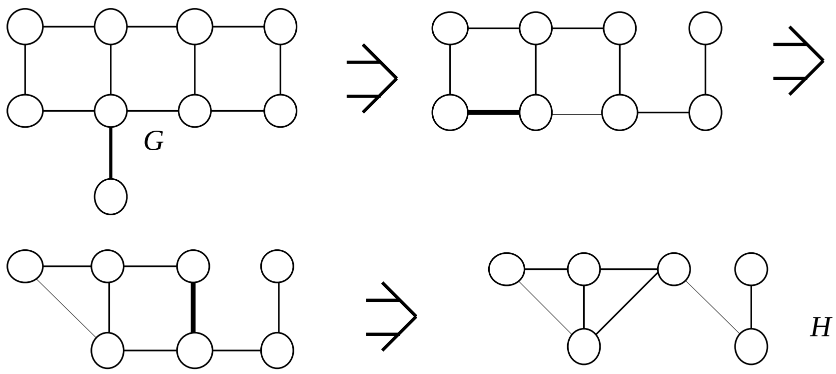

Figure 3: G is a minor of H

In many cases, the information, computed per node $i \in I$ is an element of a finite set. Then, the algorithm can be seen as a finite state tree-automata, and optimalization techniques can be applied, similar to Myhill-Nerode theory [14, 62]. (See also [48, 45, 49].)

In [64, 65] parametric problems on graphs with bounded treewidth are solved, using modifications of the technique, presented above.

For some problems (e.g. the maximum independent set problem) polynomial time algorithms are still known to exist, if the input graph is given together with a tree-decomposition of treewidth $O(\log n)$. (See e.g. [19].) For other problems, it is unknown whether such algorithms exist.

The problem whether two given graphs are isomorphic is also solvable in polynomial time, when the graphs have bounded treewidth [11, 22, 42]. The techniques are here somewhat different.

There also exist problems that remain hard when restricted to graphs with constant bounded treewidth, for instance the bandwidth problem is NP-complete for a very restricted subclass of the trees [100]. For some problems the complexity when we restrict the instances to graphs with bounded treewidth is open, like the problem to determine the pathwidth of graphs with treewidth ≤ 2 [30].

5 Graph minors

In this section, we give a short overview of some recent results on graph minors. A graph $H = (W, F)$ is a minor of a graph $G = (V, E)$, if (a graph isomorphic to) H can be obtained from G by a series of zero or more vertex deletions, edge deletions, and/or edge contractions (in arbitrary order), where an edge contraction is the operation to replace two adjacent vertices v and w by a vertex that is adjacent to all vertices that were adjacent to v or w. For an example, see figure 3.

Robertson and Seymour obtained the following deep results on graph minors

[17, 109, 115, 111, 122, 122, 116, 117, 121, 124, 123, 125, 114, 118, 119, 120, 126, 127, 128, 129, 110, 112, 113].

Theorem 5.1
For every class of graphs $\mathcal{G}$, that is closed under taking of minors, there exists a finite set of graphs, $ob(\mathcal{G})$, called the obstruction set of $\mathcal{G}$, such that for each graph G: $G \in \mathcal{G}$, if and only if there is no $H \in ob(\mathcal{G})$ that is a minor of G.

For example, the obstruction set of the planar graphs is $\{K_5, K_{3,3}\}$ [140]. Theorem 5.1 was formerly known as Wagners conjecture.

Theorem 5.2
For every graph H, there exists an $O(n^3)$ algorithm, that, given a graph G, tests whether H is a minor of G.

Theorem 5.3
For every planar graph H, there exists a constant c_H, such that for every graph G: if H is not a minor of G, then the treewidth of G is at most c_H.

The constant factor of the algorithm in theorem 5.2 is very high, making this algorithm not suitable for practical use. In [129], it is shown that one can take in 5.3 $c_H = 20^{4|V_H|+8|E_H|^5}$. From theorem 5.1 and theorem 5.2 it follows that every class of graphs, closed under minor taking, is recognizable in $O(n^3)$ time (do a minor test for each graph in the obstruction set.) Using theorem 5.1, theorem 5.3, the result of the next section, that states that for graphs with constant bounded treewidth, a tree-decomposition of constant bounded treewidth can be found in $O(n)$ time, and the fact, that with such a tree-decomposition, minor tests can be done in linear time with a procedure of the type, discussed in section 4, the following result can be derived: every class of graphs that does not contain all planar graphs and that is closed under minor taking, can be recognized in $O(n)$ time. (See also [13].)

Many applications of this theory were found by Fellows and Langston [58, 60, 61]. Note however that the constants hidden in the 'O'-notation may be quite large, and that the proof of theorem 5.1 is inherently non-constructive (in a deep mathematical sense) [66]. I.e., it is not possible in all cases to extract the obstruction set of a class of graphs $\mathcal{G}$, given a formal proof that $\mathcal{G}$ is minor closed. Thus, we may arrive in a situation where we know that a polynomial algorithm exists for the problem without knowing the algorithm itself. Also, the algorithms are recognition algorithms: they do not constuct anything (like a vertex ordering, tree-decomposition, etc.)

A technique that allows us in some cases to overcome non-constructive aspects of this theory is self-reduction, advocated by Fellows and Langston, see e.g. [21, 39, 59, 63].

Self reduction is the technique to consult a decision algorithm a number of times with different inputs in order to construct a solution for the original problem. As an example, consider the problem of finding a simple path of length at least k (k constant) in an undirected graph. (There are direct and more efficient algorithms for this problem [27, 63]; the solution here is presented only to explain the technique.) The class of graphs that do not contain such a path is closed under minor taking, and does not contain all planar graphs, so we have a linear time algorithm, deciding whether a given graph contains a simple path of length at least k. Given a graph G, we can solve the problem in $O(n \cdot e)$ time by first testing whether G contains a desired path, and then repeatedly trying to remove an edge from G, such that the resulting graph still contains a simple path of length k. When no edge can be deleted anymore, the resulting graph is precisely the desired path.

Even when we do not know the obstruction set, in several cases it is still possible to *construct* polynomial time algorithms based on minor tests (see [63]).

In some cases, obstruction sets, and hence the decision algorithms themselves are computable [12, 16, 40, 57, 62, 78, 81, 91, 103, 131, 136]. The size of the obstruction sets can grow very fast: for instance, the obstruction set of the graphs with pathwidth at most k contains at least $k!^2$ trees, each containing $\frac{5 \cdot 3^k - 1}{2}$ vertices [136]. This clearly limits the practicality of the approach described above.

Also, in some cases, linear time minor tests are possible [27, 25, 54, 63]. For instance, suppose that H is a cycle of length k. The algorithm is as follows: first make a depth-first search spanning tree $T = (V, F)$ of the input graph $G = (V, E)$. If there is a backedge between a vertex v and a predecessor w of v which is at least $k - 1$ levels above v in T, then G contains H as a minor, stop. Otherwise, construct $(\{X_v \mid v \in V\}, T = (V, F))$, with $X_v = \{v\} \cup \{w \mid w$ is a predecessor of v and differs at most $k - 2$ levels from v in $T\}$. This is a tree-decomposition of G with treewidth at most $k - 2$. Use this tree-decomposition to solve the problem in linear time. (See [63].)

6 Finding tree-decompositions

In this section we consider the problem of finding tree-decompositions, and determining the treewidth of a graph. Unfortunately, determining whether the treewidth of a given graph $G = (V, E)$ is at most a given integer k is NP-complete [6]. The latter result holds also for pathwidth [6]. The complexity of these problem has been studied for several classes of graphs. Table 1 mentions several of the known results of this type.

Polynomial time approximation algorithms with $O(\log n)$ performance ratio for treewidth, and $O(\log^2 n)$ performance ratio for pathwidth, are presented in [29]. For several classes of perfect graphs, polynomial time approximation algorithms can be found in [84]. Seymour and Thomas gave a polynomial time algorithm for the branchwidth of planar graphs [134]; this directly implies a polynomial time approximation algorithm for the treewidth of planar graphs with a performance ratio $1\frac{1}{2}$ [114].

Class	Treewidth	Pathwidth
Bounded degree	N [35]	N [101] (3)
Trees/Forests	C	P [133]
Series-parallel graphs	C	P [32]
Outerplanar graphs	C	P [32]
Halin graphs	C [143]	P [32]
k-Outerplanar graphs	C [20]	P [32]
Planar graphs	O	N [101] (3)
Chordal graphs	P (1)	N [68]
Starlike chordal graphs	P (1)	N [68]
k-Starlike chordal graphs	P (1)	P [68]
Co-chordal graphs	P [85]	P [85]
Split graphs	P (1)	P [68, 84]
Bipartite graphs	N	N
Permutation graphs	P [34]	P [34]
Circular permutation graphs	P [34]	O
Cocomparability graphs	N [6, 72]	N [6, 72]
Cographs	P [36]	P [36]
Chordal bipartite graphs	P [86]	N [35]
Interval graphs	P (2)	P (2)
Circular arc graphs	P [135]	O
Circle graphs	P [83]	N [35]

P = polynomial time solvable. C = constant, hence linear time solvable. N = NP-complete. O = Open problem. (1) The treewidth of a chordal graph equals its maximum clique size minus one. (2) The treewidth and pathwidth of an interval graphs equal its maximum clique size minus one. (3) NP-completeness is shown for vertex separation number, but this is equivalent to pathwidth.

Table 1: Complexity of Pathwidth and Treewidth on different classes of graphs

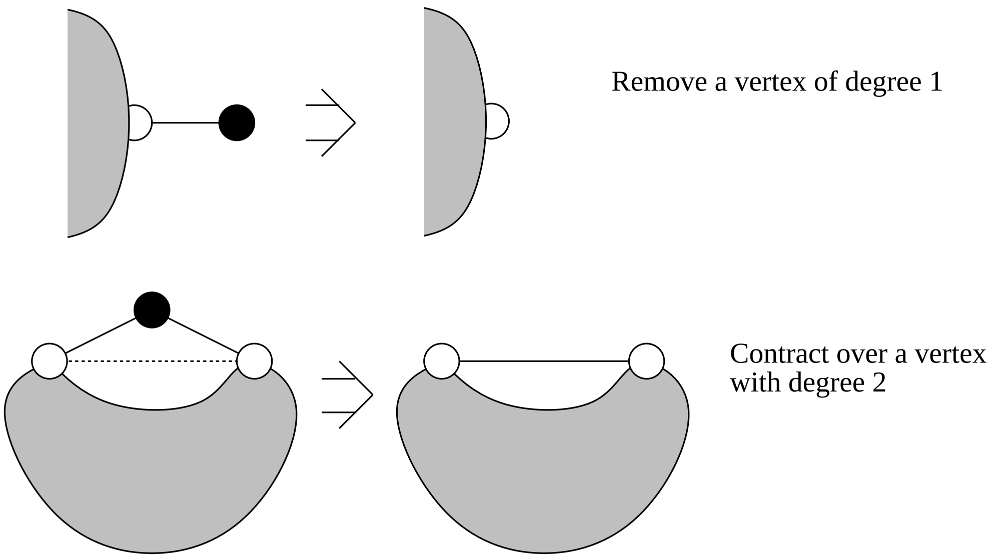

Figure 4: Rewriting a graph with treewidth ≤ 2

For constant k, polynomial time algorithms exist for the problems. The graphs with treewidth 1 are exactly the forests. Algorithms that recognize graphs with treewidth 2 and 3 in linear time, and find the corresponding tree-decompositions were described by Matousek and Thomas [97], using results from [9]. A similar algorithm (with a quite involved case analysis) for treewidth 4 was found recently by Sanders [130]. For example, the connected graphs with treewidth 2 are exactly those graphs that can be rewritten to a single vertex, using the operations shown in figure 4. For larger k, also recognition algorithms based on rewriting exist [7]. (In [7], a much larger class of problems is also shown to be solvable with these rewrite techniques.) The latter algorithms can at present, not produce a corresponding tree-decomposition of the input graph.

For arbitrary fixed k, an $O(n \log n)$ algorithm can be found, using the following result, due to Reed [108].

Theorem 6.1
For every constant k, there exists an $O(n \log n)$ algorithm, that given a graph $G = (V, E)$, either outputs that the treewidth of G is larger than k, or outputs a tree-decomposition of G with treewidth at most $3k + 2$.

Actually, the result proven by Reed has a number, larger than $3k + 2$. Minor improvements give the result stated above. The running time of this algorithm is singly exponential in k. Similar, but slower algorithms have been found by Robertson and Seymour [119] and by Lagergren [89], the latter result also has an efficient parallel variant.

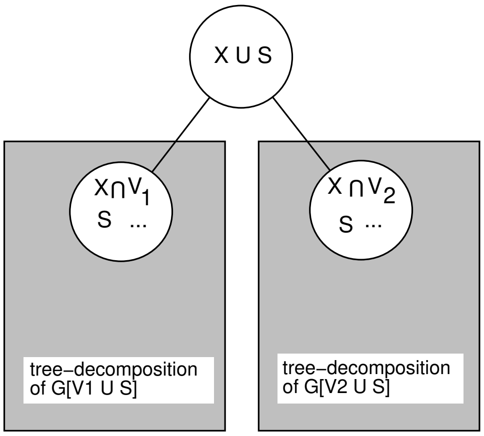

Figure 5: Illustration to approximation algorithm

These algorithms and the approximation algorithm in [29] are based on repeatedly finding separators. An 1/3-2/3 separator of a set $W \subseteq V$ in a graph $G = (V, E)$ is a set $S \subseteq V$, such that $V - S$ can be partitioned into two non-adjacent sets of vertices V_1, V_2, such that both V_1 and V_2 contain at most $2|W|/3$ vertices in W.

Each of the algorithms can be described by a recursive procedure which is called with two arguments: a graph $G' = (V', E')$ (an induced subgraph of G), and a set of vertices $X \subseteq V'$. The algorithm produces a tree-decomposition with the root node set X_r of T containing all vertices in X ($X \subseteq X_r$). It works basically as follows: When V' is 'small enough', yield a one-node tree-decomposition, the node containing all vertices in V'. Otherwise, first find a 'small' 1/3-2/3 separator S of X in G', separating $V' - S$ into V_1 and V_2. Call the procedure recursively for graph $G[V_1 \cup S]$ and set $S \cup (X \cap V_1)$, and for graph $G[V_2 \cup S]$ and set $S \cup (X \cap V_2)$. The desired tree-decomposition is obtained by taking one new node containing $X \cap S$, and connecting this node to the root nodes of the two tree-decompositions yielded by the recursive calls of the procedure (see figure 5). If the treewidth of G is at most k, then a 1/3-2/3 separator, as needed for the algorithm, exists of size at most k, and can be found, in time, linear in V', using flow techniques [119]. Starting with an arbitrary set X of size at most $3k$, it follows with induction, that each call of the procedure uses sets X of size at most $3k$, assuming the treewidth of G is at most k. ($|X \cap V_i \cup S| \leq 2|X|/3 + |S| \leq 2k + k$.) Hence, the algorithm produces in this case a tree-decomposition of treewidth less than $4k$.

Reed [108] has shown that one can also find small sized separator sets S, that do not only separate X, but also partition V' into sets of size at most 3/4 of $|V'|$. This gives a recursion depth of $O(\log n)$, and results in an $O(n \log n)$ algorithm. (The expose above is only a very rough sketch of some of the most important ideas of the algorithms. See further [29, 89, 108, 119].)

Using the algorithm of theorem 5.1, and a constant number of minor tests, it follows that the 'treewidth $\leq k$' and 'pathwidth $\leq k$' problems (for constant k) are decidable in $O(n \log n)$ time. (Use that the treewidth and pathwidth can not increase by taking minors.) However, it is also possible to obtain direct, explicit and constructive algorithms for the problems.

Both Lagergren and Arnborg [91] and Bodlaender and Kloks [31, 82] give such an algorithm, using an involved application of the technique, discussed in section 4. Independently, results of a similar nature were obtained by Abrahamson and Fellows [1]. From these results it follows that a technique of Fellows and Langston [62] can be used to compute the corresponding obstruction set. Bodlaender and Kloks [31] also discuss how in the same time bounds the path- or tree-decompositions with pathwidth or treewidth at most k can be found, if existing.

Recently, the author has found a linear time algorithm for the problems to decide whether a graph has pathwidth or treewidth at most some constant k, and if so, to find a path- or tree-decomposition with pathwidth or treewidth at most k [24]. This algorithm uses a recursion technique, and the result in [31] as essential ingredients.

A study to dynamic algorithms for graphs with small treewidth has been made by Cohen et al. [43] and recently by the author [23].

Acknowledgements

I thank Bruno Courcelle, Jens Gustedt, Ton Kloks, Mike Fellows, Detlef Seese, and Andrzej Proskurowski for useful comments on earlier versions of this tourist guide.

References

[1] K. R. Abrahamson and M. R. Fellows. Finite automata, bounded treewidth and well-quasiordering. In *Graph Structure Theory, Contemporary Mathematics vol. 147*, pages 539–564. American Mathematical Society, 1993.

[2] R. Agarwala and D. Fernandez-Baca. A polynomical-time algorithm for the phylogeny problem when the number of character states is fixed. Manuscript, 1992.

[3] S. Arnborg. Efficient algorithms for combinatorial problems on graphs with bounded decomposability – A survey. *BIT*, 25:2–23, 1985.

[4] S. Arnborg. Graph decompositions and tree automata in reasoning with uncertainty. Manuscript, to appear in Journal of Experimental and Theoretical AI, 1991.

[5] S. Arnborg. Some PSPACE-complete logic decision problems that become linear time solvable on formula graphs of bounded treewidth. Manuscript, 1991.

[6] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Alg. Disc. Meth.*, 8:277–284, 1987.

[7] S. Arnborg, B. Courcelle, A. Proskurowski, and D. Seese. An algebraic theory of graph reduction. In H. Ehrig, H. Kreowski, and G. Rozenberg, editors, *Proceedings of the Fourth Workshop on Graph Grammars and Their Applications to Computer Science*, pages 70–83. Springer Verlag, Lecture Notes in Computer Science, vol. 532, 1991. To appear in J. ACM.

[8] S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12:308–340, 1991.

[9] S. Arnborg and A. Proskurowski. Characterization and recognition of partial 3-trees. *SIAM J. Alg. Disc. Meth.*, 7:305–314, 1986.

[10] S. Arnborg and A. Proskurowski. Linear time algorithms for NP-hard problems restricted to partial k-trees. *Disc. Appl. Math.*, 23:11–24, 1989.

[11] S. Arnborg and A. Proskurowski. Canonical representations of partial 2- and 3-trees. In *Proceedings of the 2nd Scandinavian Workshop on Algorithm Theory*, pages 310–319. Springer Verlag, Lecture Notes in Computer Science, vol. 477, 1990.

[12] S. Arnborg, A. Proskurowski, and D. G. Corneil. Forbidden minors characterization of partial 3-trees. *Disc. Math.*, 80:1–19, 1990.

[13] S. Arnborg, A. Proskurowski, and D. Seese. Monadic second order logic, tree automata and forbidden minors. In E. Börger, H. Kleine Büning, M. M. Richter, and W. Schönfeld, editors, *Proceedings 4th Workshop on Computer Science Logic, CSL'90*, pages 1–16. Springer Verlag, Lecture Notes in Computer Science, vol. 533, 1991.

[14] M. W. Bern, E. L. Lawler, and A. L. Wong. Linear time computation of optimal subgraphs of decomposable graphs. *J. Algorithms*, 8:216–235, 1987.

[15] D. Bienstock. Graph searching, path-width, tree-width and related problems (a survey). *DIMACS Ser. in Discrete Mathematics and Theoretical Computer Science*, 5:33–49, 1991.

[16] D. Bienstock and N. Dean. On obstructions to small face covers in planar graphs. *J. Comb. Theory Series B*, 55:163–189, 1992.

[17] D. Bienstock, N. Robertson, P. D. Seymour, and R. Thomas. Quickly excluding a forest. *J. Comb. Theory Series B*, 52:274–283, 1991.

[18] H. L. Bodlaender. Classes of graphs with bounded treewidth. Technical Report RUU-CS-86-22, Dept. of Computer Science, Utrecht University, Utrecht, the Netherlands, 1986.

[19] H. L. Bodlaender. Dynamic programming algorithms on graphs with bounded tree-width. In *Proceedings of the 15th International Colloquium on Automata, Languages and Programming*, pages 105–119. Springer Verlag, Lecture Notes in Computer Science, vol. 317, 1988.

[20] H. L. Bodlaender. Some classes of graphs with bounded treewidth. *Bulletin of the EATCS*, 36:116–126, 1988.

[21] H. L. Bodlaender. Improved self-reduction algorithms for graphs with bounded treewidth. In *Proc. 15th Int. Workshop on Graph-theoretic Concepts in Computer Science WG'89*, pages 232–244. Springer Verlag, Lect. Notes in Computer Science, vol. 411, 1990. To appear in: Annals of Discrete Mathematics.

[22] H. L. Bodlaender. Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees. *J. Algorithms*, 11:631–643, 1990.

[23] H. L. Bodlaender. Dynamic algorithms for graphs with treewidth 2. Manuscript, 1992.

[24] H. L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. Technical Report RUU-CS-92-27, Department of Computer Science, Utrecht University, Utrecht, the Netherlands, 1992. To appear in proceedings STOC'93.

[25] H. L. Bodlaender. On disjoint cycles. In *Proceedings 17th International Workshop on Graph-Theoretic Concepts in Computer Science WG'91*, pages 230–239. Springer Verlag, Lecture Notes in Computer Science, vol. 570, 1992.

[26] H. L. Bodlaender. Complexity of path-forming games. *Theor. Comp. Sc.*, 110:215–245, 1993.

[27] H. L. Bodlaender. On linear time minor tests with depth first search. *J. Algorithms*, 14:1–23, 1993.

[28] H. L. Bodlaender, M. R. Fellows, and T. J. Warnow. Two strikes against perfect phylogeny. In *Proceedings 19th International Colloquium on Automata, Languages and Programming*, pages 273–283, Berlin, 1992. Springer Verlag, Lecture Notes in Computer Science 623.

[29] H. L. Bodlaender, J. R. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating treewidth, pathwidth, and minimum elimination tree height. In G. Schmidt and R. Berghammer, editors, *Proceedings 17th International Workshop on*

Graph-Theoretic Concepts in Computer Science WG'91, pages 1–12. Springer Verlag, Lecture Notes in Computer Science, vol. 570, 1992.

[30] H. L. Bodlaender and J. Gustedt. A conjecture on the pathwidth of k-trees. In: Proceedings AMS Summer Conference on Graph Minors, 1992. Contemp. Math. 147. In section "Open Problems", editor N. Dean, 1993.

[31] H. L. Bodlaender and T. Kloks. Better algorithms for the pathwidth and treewidth of graphs. In *Proceedings of the 18th International Colloquium on Automata, Languages and Programming*, pages 544–555. Springer Verlag, Lecture Notes in Computer Science, vol. 510, 1991.

[32] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. Manuscript. A preliminary version appeared as [31], 1993.

[33] H. L. Bodlaender and T. Kloks. A simple linear time algorithm for triangulating three-colored graphs. *J. Algorithms*, 15:160–172, 1993.

[34] H. L. Bodlaender, T. Kloks, and D. Kratsch. Treewidth and pathwidth of permutation graphs. In *Proceedings 20th International Colloquium on Automata, Languages and Programming*, pages 114–125, Berlin, 1993. Springer Verlag, Lecture Notes in Computer Science, vol. 700.

[35] H. L. Bodlaender, T. Kloks, D. Kratsch, and H. Müller, 1993. Unpublished results.

[36] H. L. Bodlaender and R. H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Disc. Meth.*, 6:181–188, 1993.

[37] R. B. Borie. *Recursively Constructed Graph Families.* PhD thesis, School of Information and Computer Science, Georgia Institute of Technology, 1988.

[38] R. B. Borie, R. G. Parker, and C. A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7:555–582, 1992.

[39] D. J. Brown, M. R. Fellows, and M. A. Langston. Nonconstructive polynomial-time decidability and self-reducibility. *Int. J. Computer Math.*, 31:1–9, 1989.

[40] R. L. Bryant, M. R. Fellows, N. G. Kinnersley, and M. A. Langston. On finding obstruction sets and polynomial-time algorithms for gate matrix layout. In *Proc. 25th Allerton Conf. on Communication, Control and Computing*, 1987.

[41] N. Chandrasekharan. *Fast Parallel Algorithms and Enumeration Techniques for Partial k-Trees.* PhD thesis, Clemson University, 1990.

[42] N. Chandrasekharan. Isomorphism testing of k-trees is in NC, for fixed k. *Inform. Proc. Letters*, 34:283–287, 1990.

[43] R. F. Cohen, S. Sairam, R. Tamassia, and J. S. Vitter. Dynamic algorithms for bounded tree-width graphs. Technical Report CS-92-19, Department of Computer Science, Brown University, 1992.

[44] D. G. Corneil and J. M. Keil. A dynamic programming approach to the dominating set problem on k-trees. *SIAM J. Alg. Disc. Meth.*, 8:535–543, 1987.

[45] B. Courcelle. The monadic second-order logic of graphs VI: On several representations of graphs by relational structures. Technical Report 89-99, Bordeaux-I University, 1989. To appear in: Discrete Applied Mathematics.

[46] B. Courcelle. Graph rewriting: an algebraic and logical approach. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, volume B*, pages 192–242, Amsterdam, 1990. North Holland Publ. Comp.

[47] B. Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Information and Computation*, 85:12–75, 1990.

[48] B. Courcelle. The monadic second-order logic of graphs V: On closing the gap between definability and recognizability. *Theor. Comp. Sc.*, 80:153–202, 1991.

[49] B. Courcelle. The monadic second-order logic of graphs VII: Graphs as relational structures. Manuscript, to appear in: Theoretical Computer Science, 1991.

[50] B. Courcelle. Graph grammars, monadic second-order logic and the theory of graph minors. *Bulletin of the EATCS*, 46:193–226, 1992. To appear in: Proceedings AMS Summer Research Conference on Graph Minors.

[51] B. Courcelle. The monadic second-order logic of graphs III: Treewidth, forbidden minors and complexity issues. *Informatique Théorique*, 26:257–286, 1992.

[52] B. Courcelle and M. Mosbah. Monadic second-order evaluations on tree-decomposable graphs. *Theor. Comp. Sc.*, 109:49–82, 1993.

[53] N. Deo, M. S. Krishnamoorty, and M. A. Langston. Exact and approximate solutions for the gate matrix layout problem. *IEEE Trans. Computer Aided Design*, 6:79–84, 1987.

[54] R. G. Downey and M. R. Fellows. Fixed-parameter tractability and completeness. Manuscript, 1991.

[55] E. S. El-Mallah and C. J. Colbourn. Partial k-tree algorithms. *Congressus Numerantium*, 64:105–119, 1988.

[56] M. R. Fellows. The Robertson-Seymour theorems: A survey of applications. *Contemporary Mathematics*, 89:1–18, 1989.

[57] M. R. Fellows, N. G. Kinnersley, and M. A. Langston. Finite-basis theorems, and a computational integrated approach to obstruction set isolation. In E. Kaltofen and S. M. Watt, editors, *Proceedings of the 3rd Conference on Computers and Mathematics*, pages 37–45, New York, 1989. Springer Verlag.

[58] M. R. Fellows and M. A. Langston. Nonconstructive advances in polynomial-time complexity. *Inform. Proc. Letters*, 26:157–162, 1987.

[59] M. R. Fellows and M. A. Langston. Fast self-reduction algorithms for combinatorial problems of VLSI design. In *Proc. 3rd Aegean Workshop on Computing*, pages 278–287. Springer Verlag, Lecture Notes in Computer Science, vol. 319, 1988.

[60] M. R. Fellows and M. A. Langston. Layout permutation problems and well-partially-ordered sets. In J. Reif, editor, *5th MIT Conf. on Advanced Research in VLSI*, pages 315–327, Cambridge, MA, 1988. Springer Verlag Lecture Notes in Computer Science 319.

[61] M. R. Fellows and M. A. Langston. Nonconstructive tools for proving polynomial-time decidability. *J. ACM*, 35:727–739, 1988.

[62] M. R. Fellows and M. A. Langston. An analogue of the Myhill-Nerode theorem and its use in computing finite-basis characterizations. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 520–525, 1989.

[63] M. R. Fellows and M. A. Langston. On search, decision and the efficiency of polynomial-time algorithms. In *Proceedings of the 21rd Annual Symposium on Theory of Computing*, pages 501–512, 1989.

[64] D. Fernández-Baca and G. Slutzki. Solving parametric problems on trees. *J. Algorithms*, 10:381–402, 1989.

[65] D. Fernández-Baca and G. Slutzki. Parametic problems on graphs of bounded treewidth. In O. Nurmi and E. Ukkonen, editors, *Proceedings 3rd Scandinavian Workshop on Algorithm Theory*, pages 304–316. Springer Verlag, Lecture Notes in Computer Science, vol. 621, 1992.

[66] H. Friedman, N. Robertson, and P. D. Seymour. The metamathematics of the graph minor theorem. *Contemporary Mathematics*, 65:229–261, 1987.

[67] D. Granot and D. Skorin-Kapov. On some optimization problems on k-trees and partial k-trees. Manuscript, to appear in Discrete Appl. Math., 1988.

[68] J. Gustedt. Path width for chordal graphs is NP-complete. Technical Report 221/1989, Technical University Berlin, 1989. To appear in Discr. Appl. Math.

[69] A. Habel. Graph-theoretic properties compatible with graph derivations. In J. van Leeuwen, editor, *Proceedings 14th International Workshop on Graph-Theoretic Concepts in Computer Science WG'88*, pages 11–29. Springer Verlag, Lecture Notes in Computer Science, vol. 344, 1988.

[70] A. Habel and H. J. Kreowski. May we introduce to you: hyperedge replacement. In H. Ehrig, M. Nagl, and A. Rosenberg, editors, *Proc. Graph-Grammars and their Applications to Computer Science '86*, pages 15–26. Springer Verlag, Lect. Notes in Comp. Science vol. 291, 1987.

[71] A. Habel and H.-J. Kreowski. Filtering hyperedge-replacement languages through compatible properties. In *Proceedings 15th International Workshop on Graph-Theoretic Concepts in Computer Science WG'89*, 1990.

[72] M. Habib and R. H. Möhring. Treewidth of cocomparability graphs and a new order-theoretic parameter. Technical Report 336/1992, Fachbereich Mathematik, Technische Universität Berlin, 1992.

[73] E. Hare, S. Hedetniemi, R. Laskar, K. Peters, and T. Wimer. Linear-time comptability of combinatorial problems on generalized-series-parallel graphs. In D. S. Johnson, T. Nishizeki, A. Nozaki, and H. S. Wilf, editors, *Proc. of the Japan-US Joint Seminar on Discrete Algorithms and Complexity*, Orlando, Florida, 1987. Academic Press, Inc.

[74] W. Hohberg and R. Reischuk. A framework to design algorithms for optimization problems on graphs. Preprint, April 1990.

[75] K. Jansen and P. Scheffler. Generalized coloring for tree-like graphs. In *Proceedings 18th International Workshop on Graph-Theoretic Concepts in Computer Science WG'92*, pages 50–59, Berlin, 1993. Springer Verlag, Lecture Notes in Computer Science, vol. 657.

[76] D. S. Johnson. The NP-completeness column: An ongoing guide. *J. Algorithms*, 6:434–451, 1985.

[77] D. S. Johnson. The NP-completeness column: An ongoing guide. *J. Algorithms*, 8:285–303, 1987.

[78] Y. Kajitani, A. Ishizuka, and S. Ueno. Characterization of partial 3 trees in terms of 3 structures. *Graphs and Combinatorics*, 2:233–246, 1986.

[79] S. Kannan and T. Warnow. Inferring evolutionary history from DNA sequences. In *Proceedings of the 31rd Annual Symposium on Foundations of Computer Science*, pages 362–371, 1990.

[80] S. Kannan and T. Warnow. Triangulating 3-colored graphs. *SIAM J. Disc. Meth.*, 5:249–258, 1992.

[81] N. G. Kinnersley. *Obstruction Set Isolation for Layout Permutation Problems.* PhD thesis, Washington State University, May 1989.

[82] T. Kloks. *Treewidth.* PhD thesis, Utrecht University, Utrecht, the Netherlands, 1993.

[83] T. Kloks. Treewidth of circle graphs. Technical Report RUU-CS-93-12, Department of Computer Science, Utrecht University, Utrecht, 1993.

[84] T. Kloks and H. Bodlaender. Approximating treewidth and pathwidth of some classes of perfect graphs. In *Proceedings Third International Symposium on Algorithms and Computation, ISAAC'92*, pages 116–125, Berlin, 1992. Springer Verlag, Lecture Notes in Computer Science, vol. 650.

[85] T. Kloks, H. Bodlaender, H. Müller, and D. Kratsch. Computing treewidth and minimum fill-in: All you need are the minimal separators. To appear in: proceedings 1st European Symposium on Algorithms, ESA'93, 1993.

[86] T. Kloks and D. Kratsch. Treewidth of chordal bipartite graphs. In P. Enjalbert, A. Finkel, and K. W. Wagner, editors, *Proceedings Symp. Theoretical Aspects of Computer Science, STACS'93*, pages 80–89, Berlin, 1993. Springer Verlag, Lecture Notes in Computer Science, vol. 665.

[87] E. Korach and N. Solel. Linear time algorithm for minimum weight Steiner tree in graphs with bounded treewidth. Manuscript, 1990.

[88] A. Kornai and Z. Tuza. Narrowness, pathwidth, and their application in natural language processing. Manuscript. Submitted Disc. Appl. Math., 1990.

[89] J. Lagergren. Efficient parallel algorithms for tree-decomposition and related problems. In *Proceedings of the 31rd Annual Symposium on Foundations of Computer Science*, pages 173–182, 1990.

[90] J. Lagergren. *Algorithms and Minimal Forbidden Minors for Tree-decomposable Graphs.* PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 1991.

[91] J. Lagergren and S. Arnborg. Finding minimal forbidden minors using a finite congruence. In *Proceedings of the 18th International Colloquium on Automata, Languages and Programming*, pages 533–543. Springer Verlag, Lecture Notes in Computer Science, vol. 510, 1991.

[92] S. J. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *The Journal of the Royal Statistical Society. Series B (Methodological)*, 50:157–224, 1988.

[93] C. Lautemann. Efficient algorithms on context-free graph languages. In *Proceedings of the 15th International Colloquium on Automata, Languages and Programming*, pages 362–378. Springer Verlag, Lect. Notes in Comp. Sc. 317, 1988.

[94] S. Mahajan and J. G. Peters. Regularity and locality in k-terminal graphs. Manuscript, 1990.

[95] E. Mata-Montero. Resilience of partial k-tree networks with edge and node failures. *Networks*, 21:321–344, 1991.

[96] J. Matousěk and R. Thomas. On the complexity of finding iso- and other morphisms for partial k-trees. Manuscript, to appear in: Topological, Algebraical, and Combinatorial Structures, J. Nesetril, ed., North-Holland, 1988.

[97] J. Matousěk and R. Thomas. Algorithms finding tree-decompositions of graphs. *J. Algorithms*, 12:1–22, 1991.

[98] F. R. McMorris, T. Warnow, and T. Wimer. Triangulating colored graphs. In proceedings SODA'92, to appear in SIAM J. Disc. Math., 1991.

[99] R. H. Möhring. Graph problems related to gate matrix layout and PLA folding. In E. Mayr, H. Noltemeier, and M. Sysło, editors, *Computational Graph Theory, Comuting Suppl. 7*, pages 17–51. Springer Verlag, 1990.

[100] B. Monien. The bandwidth minimization problem for caterpillars with hair length 3 is NP-complete. *SIAM J. Alg. Disc. Meth.*, 7:505–512, 1986.

[101] B. Monien and I. H. Sudborough. Min cut is NP-complete for edge weighted trees. *Theor. Comp. Sc.*, 58:209–229, 1988.

[102] M. H. Mosbah. *Constructions d'Algorithmes Pour les Graphes Structurés par des Méthodes Algébriques et Logiques.* PhD thesis, Université Bordeaux-I, 1992.

[103] R. Motwani, A. Raghunathan, and H. Saran. Constructive results from graph minors: Linkless embeddings. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, pages 398–407, 1988.

[104] A. Proskurowski. Separating subgraphs in k-trees: Cables and caterpillars. *Disc. Math.*, 49:275–285, 1984.

[105] A. Proskurowski. Maximal graphs of pathwidth k or searching a partial k-caterpillar. Technical Report CIS-TR-89-17, Dept. of Computer and Information Science, University of Oregon, 1989.

[106] A. Proskurowski and M. M. Sysło. Efficient computations in tree-like graphs. Technical Report 235, Mathematik, Techn. Univ. Berlin, 1989.

[107] V. Radhakrishnan, H. B. Hunt III, and R. E. Stearns. Efficient algorithms for solving systems of linear equations and path problems. Technical Report 91-21, Dept. of Computer Science, SUNY Albany, 1991.

[108] B. Reed. Finding approximate separators and computing tree-width quickly. In *Proceedings of the 24th Annual Symposium on Theory of Computing*, pages 221–228, 1992.

[109] N. Robertson and P. D. Seymour. Graph minors. I. Excluding a forest. *J. Comb. Theory Series B*, 35:39–61, 1983.

[110] N. Robertson and P. D. Seymour. Generalizing Kuratowskis theorem. *Congressus Numerantium*, 45:129–138, 1984.

[111] N. Robertson and P. D. Seymour. Graph minors. III. Planar tree-width. *J. Comb. Theory Series B*, 36:49–64, 1984.

[112] N. Robertson and P. D. Seymour. Graph width and well-quasi ordering: a survey. In J. A. Bondy and U. S. R. Murty, editors, *Progress in Graph Theory*, pages 399–406, Toronto, 1984. Academic Press.

[113] N. Robertson and P. D. Seymour. Graph minors — a survey. In I. Anderson, editor, *Surveys in Combinatorics*, pages 153–171. Cambridge Univ. Press, 1985.

[114] N. Robertson and P. D. Seymour. Graph minors. XI. Distance on a surface. Manuscript, 1985.

[115] N. Robertson and P. D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms*, 7:309–322, 1986.

[116] N. Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *J. Comb. Theory Series B*, 41:92–114, 1986.

[117] N. Robertson and P. D. Seymour. Graph minors. VI. Disjoint paths across a disc. *J. Comb. Theory Series B*, 41:115–138, 1986.

[118] N. Robertson and P. D. Seymour. Graph minors. XII. Excluding a non-planar graph. Manuscript, 1986.

[119] N. Robertson and P. D. Seymour. Graph minors. XIII. The disjoint paths problem. Manuscript, 1986.

[120] N. Robertson and P. D. Seymour. Graph minors. XIV. Taming a vortex. Manuscript, 1987.

[121] N. Robertson and P. D. Seymour. Graph minors. VII. Disjoint paths on a surface. *J. Comb. Theory Series B*, 45:212–254, 1988.

[122] N. Robertson and P. D. Seymour. Graph minors. IV. Tree-width and well-quasi-ordering. *J. Comb. Theory Series B*, 48:227–254, 1990.

[123] N. Robertson and P. D. Seymour. Graph minors. IX. Disjoint crossed paths. *J. Comb. Theory Series B*, 49:40–77, 1990.

[124] N. Robertson and P. D. Seymour. Graph minors. VIII. A Kuratowski theorem for general surfaces. *J. Comb. Theory Series B*, 48:255–288, 1990.

[125] N. Robertson and P. D. Seymour. Graph minors. X. Obstructions to tree-decomposition. *J. Comb. Theory Series B*, 52:153–190, 1991.

[126] N. Robertson and P. D. Seymour. Graph minors. XV. Etending an embedding. Manuscript, 1991.

[127] N. Robertson and P. D. Seymour. Graph minors. XVI. Giant steps. Manuscript, 1991.

[128] N. Robertson and P. D. Seymour. Graph minors. XVII. Excluding a non-planar graph. Manuscript, 1991.

[129] N. Robertson, P. D. Seymour, and R. Thomas. Quickly excluding a planar graph. Technical Report TR89-16, DIMACS, 1989.

[130] D. P. Sanders. On linear recognition of tree-width at most four. Manuscript, 1992.

[131] A. Satyanarayana and L. Tung. A characterization of partial 3-trees. *Networks*, 20:299–322, 1990.

[132] P. Scheffler. *Die Baumweite von Graphen als ein Maß für die Kompliziertheit algorithmischer Probleme.* PhD thesis, Akademie der Wissenschaften der DDR, Berlin, 1989.

[133] P. Scheffler. A linear algorithm for the pathwidth of trees. In R. Bodendiek and R. Henn, editors, *Topics in combinatorics and graph theory*, pages 613–620, Heidelberg, 1990. Physica-Verlag.

[134] P. D. Seymour and R. Thomas. Call routing and the ratcatcher. Manuscript, 1990.

[135] R. Sundaram, K. Sher Singh, and C. Pandu Rangan. Treewidth of circular-arc graphs. Manuscript, to appear in SIAM J. Disc. Math., 1991.

[136] A. Takahashi, S. Ueno, and Y. Kajitani. Minimal acyclic forbidden minors for the family of graphs with bounded path-width. In *SIGAL 91-19-3, IPSJ*, 1991. To appear in: *Annals of discrete mathematics* (Proceedings of 2nd Japan conference on graph theory and combinatorics, 1990).

[137] J. Telle and A. Proskurowski. Efficient sets in partial k-trees. Technical report, Department of Computer and Information Science, University of Oregon, 1991.

[138] L. C. van der Gaag. *Probability-Based Models for Plausible Reasoning*. PhD thesis, University of Amsterdam, 1990.

[139] J. van Leeuwen. Graph algorithms. In *Handbook of Theoretical Computer Science, A: Algorithms and Complexity Theory*, pages 527–631, Amsterdam, 1990. North Holland Publ. Comp.

[140] K. Wagner. Über eine Eigenshaft der ebenen Complexe. *Math. Ann.*, 14:570–590, 1937.

[141] M. Wiegers. The k-section of treewidth restricted graphs. In B. Rovan, editor, *Proceedings Conference on Mathematical Foundations of Computer Science MFCS'90*, pages 530–537, Berlin, 1990. Springer Verlag, Lecture Notes in Computer Science, vol. 452.

[142] T. V. Wimer. Linear algorithms for the dominating cycle problems in series-parallel graphs, 2-trees and Halin graphs. *Congressus Numerantium*, 56, 1987.

[143] T. V. Wimer. *Linear Algorithms on k-Terminal Graphs*. PhD thesis, Dept. of Computer Science, Clemson University, 1987.

[144] T. V. Wimer, S. T. Hedetniemi, and R. Laskar. A methodology for constructing linear graph algorithms. *Congressus Numerantium*, 50:43–60, 1985.

[145] X. Zhou, S. Nakano, H. Suzuki, and T. Nishizeki. An efficient algorithm for edge-coloring series-parallel multigraphs. In I. Simon, editor, *Proceedings LATIN'92*, pages 516–529. Springer Verlag, Lecture Notes in Computer Science, vol. 583, 1992.