Treewidth of Circle Graphs

T. Kloks

RUU-CS-93-12
March 1993

Utrecht University

Department of Computer Science

Padualaan 14, P.O. Box 80.089,
3508 TB Utrecht, The Netherlands,
Tel. : ... + 31 - 30 - 531454

Treewidth of Circle Graphs

T. Kloks

Technical Report RUU-CS-93-12
March 1993

Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht
The Netherlands

ISSN: 0924–3275

Treewidth of Circle Graphs

T. Kloks *
Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht, The Netherlands

Abstract

In this paper we show that the treewidth of a circle graph can be computed in polynomial time. A circle graph is a graph that is isomorphic to the intersection graph of a finite collection of chords of a circle. The TREEWIDTH problem can be viewed upon as the problem of finding a chordal embedding of the graph that minimizes the maximum clique size. Our algorithm to determine the treewidth of a circle graph can be implemented to run in $O(n^3)$ time, where n is the number of vertices of the graph.

1 Introduction

Consider a set of n chords of a circle. Associate with this set an undirected graph as follows. The vertex set is the set of chords and two vertices are adjacent if and only if the corresponding chords intersect. Such a graph is called a *circle graph* and we call a set of chords representing the graph a *circle model*. In this paper we do not distinguish between a circle graph and the circle model, i.e., we assume that we have a circle model of the graph. If the circle model is not given, it can be found in $O(n^2)$ time [9, 15], as reported in [6].

It is interesting to note that some important problems remain NP-complete when restricted to circle graphs. These problems include for example the CHROMATIC NUMBER problem [12], the COCHROMATIC NUMBER problem [10, 19] and the ACHROMATIC NUMBER problem [2]. There exists a heuristic for coloring circle graphs with performance guarantee of $O(\log n)$ [18]. On the other hand, some NP-complete problems are solvable in polynomial time, when restricted to circle graphs, for example the MAXIMUM INDEPENDENT SET, which can be solved in $O(n^2)$ time [17]. For more information on circle graphs and related classes of graphs we refer to [11, 6].

*This author is supported by the Foundation for Computer Science (S.I.O.N.) of the Netherlands Organization for Scientific Research (N.W.O.). Email: ton@cs.ruu.nl.

The treewidth of a graph is the minimum of the maximum cliquesize minus one over all chordal embeddings of a graph. Since so many problems become solvable in polynomial time when restricted to the class of graphs with bounded treewidth, it is of importance to find a chordal embedding of a graph with a small clique size. Since the problem is NP-complete in general it is of interest to find fast algorithms for special classes of graphs. The TREEWIDTH problem can be solved in polynomial time for example for cographs [5], permutation graphs [4, 13], chordal bipartite graphs [14, 13], circular arc graphs [16], cotriangulated graphs [13] and for the class of graphs with bounded treewidth [3]. The treewidth problem remains NP-complete when restricted to bipartite and cobipartite graphs [1]. For more information on treewidth we refer to [13].

In this paper we give a simple and efficient algorithm to determine the treewidth of circle graphs. To illustrate the simplicity of the algorithm, and to whet the reader's appetite, we describe the algorithm here. Consider the circle model. Go around the circle in clockwise order and place a new vertex between every two consecutive end vertices of chords. Let Z be the set of these new vertices. Consider the polygon $\mathcal{P}$ with vertex set Z, and let T be a triangulation of this polygon. For a triangle in this triangulation define the weight as the number of chords in the circle model that cross the triangle. The weight of the triangulation T is the maximum weight of the triangles. The treewidth of the circle graph is the minimum weight minus one over all triangulations of the polygon $\mathcal{P}$. It is not hard to see that, using dynamic programming, the treewidth can be computed in $O(n^3)$ time.

Since the class of permutation graphs is properly contained in the class of circle graphs, our result generalizes some results of [4] where an $O(nk^2)$ algorithm is given for the treewidth (and pathwidth) of permutation graphs, where k is the treewidth of the graph.

2 Preliminaries

In this section we start with some necessary definitions and lemmas.

Definition 2.1 *A* circle graph *is a graph for which one can associate for each vertex a chord of a circle such that two vertices are adjacent if and only if the corresponding chords have a nonempty intersection.*

Without loss of generality we can assume that no two chords share an end vertex. A set of chords of a circle such that the graph is isomorphic with the intersection graph is called a *circle model* for the graph. Throughout this paper we identify a circle graph and a circle model of the graph, i.e., we assume that we have a circle model.

Definition 2.2 *A graph is* chordal *if it has no induced chordless cycle of length at least four.*

Definition 2.3 *A* triangulation *of a graph* G *is a graph* H *with the same vertex set as* G *such that* G *is a subgraph of* H *and such that* H *is chordal.*

There are two problems concerned with triangulations of graphs that have drawn much attention because of the large number of applications. One is called the MINIMUM FILL-IN problem. In this case one tries to find a triangulation of the graph with a minimum number of edges. The other problem is the TREEWIDTH problem. In this case the problem is to find a triangulation H such that the maximum number of vertices in a clique of H, $\omega(H)$, is as small as possible. The treewidth is this number of vertices in the maximum clique of H minus one. Both problems are NP-complete [1, 20]. In this paper we concentrate on finding the treewidth of circle graphs.

One of the main tools in this paper is a method to locate all minimal vertex separators in circle graphs quickly.

Definition 2.4 *Let* $G = (V, E)$ *be a graph. A subset* $S \subseteq V$ *of vertices is an* a, b-separator *for non adjacent vertices* a *and* b *if the removal of* S *separates* a *and* b *in distinct connected components. If no proper subset of* S *is an* a, b*-separator then* S *is a* minimal a, b-separator. *A* minimal separator S *is a subset* S *of vertices for which there are nonadjacent vertices* a *and* b *such that* S *is a minimal* a, b*-separator.*

The following characterization of chordal graphs was found by Dirac [7].

Lemma 2.1 *A graph* G *is chordal if and only if every minimal vertex separator induces a complete subgraph.*

If $G = (V, E)$ is a graph and S is a subset of vertices then we write $G[S]$ for the subgraph of G induced by S. We use the following theorem which appeared in [4, 13].

Theorem 2.1 *Let* G *be a graph with treewidth* k. *There exists a triangulation of* G *into a chordal graph* H *such that the following three statements hold:*

1. $\omega(H) = k + 1$.
2. *If* a *and* b *are nonadjacent vertices in* H *then every minimal* a, b*-separator in* H *is also a minimal* a, b*-separator in* G.
3. *If* S *is a minimal separator in* H *and* C *is the vertex set of a connected component in* $H[V - S]$ *then* C *induces also a connected component in* $G[V - S]$.

Definition 2.5 *We call a triangulation of which the existence is guaranteed by Theorem 2.1 a* minimal triangulation.

3 Scanlines

As mentioned before, we assume that no two chords of the circle model share an end vertex.

Definition 3.1 *Place new points on the circle as follows. Go around the circle in clockwise order. Between every two consecutive end vertices of chords, place a new vertex. These new vertices are called* scanline vertices.

If n is the number of vertices in the circle graph then there are $2n$ scanline vertices. We denote the set of scanline vertices by Z.

Definition 3.2 *A* scanline *is a chord of the circle of which the end vertices are scanline vertices.*

Definition 3.3 *Two scanlines* cross *if they have a nonempty intersection but no end vertex in common.*

Definition 3.4 *Given two non crossing chords s_1 and s_2. A scanline s is* between *s_1 and s_2 if every path from an end vertex of s_1 to an end vertex of s_2 along the circle passes through an end vertex of s.*

If a and b are nonadjacent vertices of the circle graph then the corresponding chords in the circle model do not cross. Take a scanline s which is between the chords of a and b. Clearly, the set of vertices, corresponding to chords that cross s, is an a, b-separator. The following lemma is a generalization of a result in [4].

Lemma 3.1 *Let G be a circle graph and let a and b be non adjacent vertices. For every minimal a, b-separator S there exists a scanline s between a and b such that the chords corresponding to vertices of S are exactly the chords crossing s.*

Proof. The proof is basically the same as in [4]. □

Corollary 3.1 *There are at most $O(n^2)$ minimal vertex separators in a circle graph.*

4 Components and realizers

Let $G = (V, E)$ be a circle graph. Consider a circle model for G with the set Z of scanline vertices.

Definition 4.1 *Let $Y \subseteq Z$ be a set of scanline vertices with at least three elements. Consider the convex polygon $\mathcal{P}(Y)$ with vertex set Y. The* component $G(Y)$ *is the subgraph of G induced by the set of vertices corresponding with chords in the circle model which have a non empty intersection with the interior region of $\mathcal{P}(Y)$.*

Hence the edges of the polygon $\mathcal{P}(Y)$ are scanlines. Notice that if $Y = Z$ then $G(Y)$ is simply the graph G.

Definition 4.2 *Let Y be a set of at least three scanline vertices and consider the component $G(Y)$. For each scanline that is an edge of the polygon $\mathcal{P}(Y)$, add edges between vertices of $G(Y)$ of which the corresponding chords cross that scanline. In this way we obtain the* realizer $R(Y)$ *of the component $G(Y)$.*

Hence each component is a subgraph of its realizer. We mention here that if Y is a set of three scanline vertices then

1. each chord corresponding with a vertex of $G(Y)$ intersects exactly two edges of $\mathcal{P}(Y)$ and hence
2. $R(Y)$ is a clique.

Lemma 4.1 *If $G(Y)$ is a component then the realizer $R(Y)$ is a circle graph.*

Proof. Let $s_1, s_2, \ldots, s_t$ be the scanlines which are the edges of the polygon $\mathcal{P}(Y)$. Consider these scanlines one by one.

Let $c_1, c_2, \ldots, c_\ell$ be the chords that cross some scanline s_k. Call the end vertices of c_i, a_i and b_i. Choose a_i and b_i such that, when going along the chord from a_i to b_i first the scanline s_k is crossed before the interior region of $\mathcal{P}(Y)$ is entered. Rearrange the order of the vertices $a_1, \ldots, a_\ell$ on the circle such that, afterwards, every pair of chords c_i and c_j cross. Notice that this only adds edges in the component between vertices with corresponding chords in $\{c_1, \ldots, c_\ell\}$.

In this way we obtain a circle model for the realizer. □

We identify the realizer $R(Y)$ with a circle model for $R(Y)$ obtained as in the proof of Lemma 4.1.

Definition 4.3 *Let $G(Y)$ be a component with realizer $R(Y)$. A scanline s in the circle model for $R(Y)$ is* Y-nice *if the end vertices of s are elements of Y.*

We now state one of our main results.

Lemma 4.2 *Let $R(Y)$ be a realizer of a component $G(Y)$. Let a and b be two non adjacent vertices in $R(Y)$ and let S be a minimal a, b-separator in $R(Y)$. Then there is a Y-nice scanline s such that S consists of the vertices corresponding with the chords that cross s.*

Proof. Consider the circle model for $R(Y)$. Since a and b are not adjacent we know that there is a scanline s (with end vertices in Z) between the chords of a and b such that the set of chords crossing s corresponds to S. Choose such a scanline s with a minimum number of end vertices that are not in Y. If both end vertices of s are elements of Y then s is Y-nice. Assume this is not the case. Then s crosses with

at least one scanline s' which is an edge of the polygon $\mathcal{P}(Y)$. Let the end vertices of s be x and y chosen in such a way that, if we traverse s from x to y, then we first cross s' before entering the region of $\mathcal{P}(Y)$.

Let α and β be the end vertices of s' chosen such that α is on the same side of s as the chord a, and β is on the same side of s as the chord b.

Let s^* be the scanline with end vertices y and α and let s^{**} be the scanline with end vertices y and β.

Since a and b are non adjacent in $R(Y)$ the corresponding chords of a and b do not both cross the scanline s'. Assume that the chord of a does *not* cross with s'. We now consider two cases.

b and s' do not cross. Then s^* and s^{**} are both scanlines between a and b. Let S^* and S^{**} be the corresponding separators. We claim that either $S^* \subseteq S$ or $S^{**} \subseteq S$. This can be seen as follows. Assume there is a chord p in the realizer crossing with s^* but not with s and a chord q in the realizer crossing with s^{**} but not with s. Then p and q must both cross with s'. But this is a contradiction since p and q cannot cross without also crossing s.

b and s' do cross. In this case s^* is a scanline between a and b. We claim that $S^* \subseteq S$. Assume there is a chord p in $R(Y)$ that crosses with s^* but not with s. Then p and b cannot cross. But this is a contradiction, since both p and b cross with s'.

This shows that either s^* or s^{**} is a scanline between a and b of which the corresponding separator is a subset of the separator of s. But both s^* and s^{**} both have one more end vertex in Y. This proves the theorem. □

Definition 4.4 *A component $G(Y)$ is k-feasible if the realizer $R(Y)$ has treewidth at most k.*

In other words, the component $G(Y)$ is k-feasible if and only if there is a triangulation of the component such that each clique has at most $k+1$ vertices and such that for every scanline which is an edge of $\mathcal{P}(Y)$ the set of vertices that cross that scanline induce a clique in the triangulation.

Lemma 4.3 *Let Y be a set of scanline vertices with at least three elements such that the component $G(Y)$ has at least $k+2$ vertices. Then the following two statements hold.*

1. *If $G(Y)$ is k-feasible then there is a Y-nice scanline, dividing the polygon $\mathcal{P}(Y)$ in two new polygons with vertex sets, say, Y_1 and Y_2, such that the components $G(Y_1)$ and $G(Y_2)$ both have less vertices than $G(Y)$ and such that both $G(Y_1)$ and $G(Y_2)$ are k-feasible, and*

2. *if there is a* Y*-nice scanline, dividing the polygon* $\mathcal{P}(Y)$ *in two new smaller polygons with vertex sets* Y_1 *and* Y_2*, such that the components* $G(Y_1)$ *and* $G(Y_2)$ *are both* k*-feasible, then* $G(Y)$ *is also* k*-feasible.*

Proof. Assume that $G(Y)$ is k-feasible. Consider a minimal triangulation H of $R(Y)$. Since the number of vertices is at least $k+2$ there must be a pair of non adjacent vertices a and b in H. Consider a minimal a,b-separator S in H. By Theorem 2.1 this is also a minimal a,b-separator in G. Since H is chordal S induces a clique in H. By Lemma 4.2 there is a Y-nice scanline s corresponding with S. s divides the polygon $\mathcal{P}(Y)$ into two new polygons. Let Y_1 and Y_2 be the vertex sets of these two new polygons.

We may assume that $G(Y_1)$ contains the chord corresponding with a and $G(Y_2)$ contains the chord corresponding with b. Let C_a and C_b be the vertex sets of $G(Y_1)$ and $G(Y_2)$ respectively. Then it follows that the induced subgraphs $H[C_a]$ and $H[C_b]$ are triangulations of $R(Y_1)$ and $R(Y_2)$, and hence $G(Y_1)$ and $G(Y_2)$ are k-feasible. Since a is not in $G(Y_2)$ and b not in $G(Y_1)$ it follows that both these components have less vertices than $G(Y)$.

Assume that there is a Y-nice scanline s dividing the polygon $\mathcal{P}(Y)$ in two new smaller polygons with vertex sets Y_1 and Y_2, such that the components $G(Y_1)$ and $G(Y_2)$ are both k-feasible. Consider triangulations H_1 and H_2 of $R(Y_1)$ and $R(Y_2)$ respectively. Let S be the set of vertices corresponding with chords that cross s. Since S induces a clique in H_1 and in H_2, it follows that we can obtain a triangulation H of $R(Y)$ by identifying the vertices of S is H_1 and H_2. This shows that $G(Y)$ is k-feasible. □

Definition 4.5 *Let* $\mathcal{P}$ *be a polygon with* m *vertices. A* triangulation *of* $\mathcal{P}$ *is a set of* $m-3$ *non crossing diagonals in* $\mathcal{P}$ *that divide the interior of* $\mathcal{P}$ *in* $m-2$ *triangles.*

Definition 4.6 *Let* Y *be a set of at least three scanline vertices. Consider a triangulation* T *of* $\mathcal{P}(Y)$*. The* weight *of a triangle is the number of chords in the circle model that have a non empty intersection with the triangle. The* weight of the triangulation, $w(T)$*, is the maximum weight of a triangle.*

We can now state our main result.

Theorem 4.1 *Let* Y *be a set of at least three scanline vertices. A component* $G(Y)$ *is* k*-feasible if and only if there is a triangulation* T *with weight at most* $k+1$*.*

Proof. First assume that $G(Y)$ is k-feasible. If $G(Y)$ has at most $k+1$ vertices, then *any* triangulation of $\mathcal{P}(Y)$ has weight at most $k+1$. We proceed with induction on the number of vertices of $G(Y)$. Assume $G(Y)$ has more than $k+1$ vertices. By the first part of Lemma 4.3 there is a Y-nice scanline which divides the polygon $\mathcal{P}(Y)$ in two new polygons $\mathcal{P}(Y_1)$ and $\mathcal{P}(Y_2)$ such that the components $G(Y_1)$ and

$G(Y_2)$ both have less vertices than $G(Y)$ and such that both $G(Y_1)$ and $G(Y_2)$ are k-feasible. By induction there are triangulations T_1 of $\mathcal{P}(Y_1)$ and T_2 of $\mathcal{P}(Y_2)$ both with weight at most $k+1$. Then, clearly, $T = T_1 \cup T_2$ is a triangulation of $\mathcal{P}(Y)$.

Now assume T is a triangulation of $\mathcal{P}(Y)$ with weight at most $k+1$. If Y has only three vertices, $G(Y)$ has at most $k+1$ vertices, and hence $G(Y)$ is k-feasible. We now proceed with induction on the number of vertices of Y. Take any diagonal of T. This divides the polygon into two smaller polygons with vertex sets Y_1 and Y_2 say. Since there are triangulations of $\mathcal{P}(Y_1)$ and of $\mathcal{P}(Y_2)$ with weight at most $k+1$ we may conclude that both $G(Y_1)$ and $G(Y_2)$ are k-feasible. But then by the second part of lemma 4.3 also $G(Y)$ is k-feasible. □

5 Algorithm

In this section we describe an algorithm to find the treewidth of a circle graph.

Theorem 5.1 *Given a circle graph G with n vertices. There exists an $O(n^3)$ algorithm to determine the treewidth of G.*

Proof. First compute a circle model for G. As mentioned earlier this step can be performed in $O(n^2)$ time. We may assume that we can decide whether two chords cross in $O(1)$ time.

Clearly we may assume that $n > 1$. Determine a set of scanline vertices Z. Since $n > 1$, Z has at least four vertices, and hence the polygon $\mathcal{P}(Z)$ is well defined. The algorithm we describe finds a triangulation with minimal weight for $\mathcal{P}(Z)$.

First, for each of the scanlines compute the number of chords that cross the scanline. Since there are $O(n^2)$ scanlines, and the test if a scanline and a chord cross can be performed in $O(1)$ time, this step costs $O(n^3)$ time.

Use dynamic programming to find an optimal triangulation for $\mathcal{P}(Z)$. Let the scanline vertices be $s_0, s_2, \ldots, s_{\ell-1}$ ordered clockwise. Let $P(i,t)$ be the polygon defined by $s_i, \ldots, s_{i+t-1}$, where indices are to be taken modulo ℓ. We define $w(i,t)$ as the minimum weight of a triangulation of the polygon $P(i,t)$. Let $c(i,j)$ be the number of chords crossing the scanline with end vertices s_i and s_j. Then $w(i,t)$ can be determined in $O(n^3)$ time using the following. Set all $w(i,2)$ equal to 0. For $t = 3, \ldots, \ell$, compute for all i:

$$w(i,t) = \min_{2 \le j < t} \left(\max \left(w(i,j),\ w(i+j-1, t-j+1),\ F(i,j) \right) \right)$$

$$\text{where } F(i,j) = \frac{c(i,i+j-1) + c(i+j-1, i+t-1) + c(i,i+t-1)}{2}$$

Correctness follows from the fact that each chord crossing a triangle intersects exactly two sides of the triangle. The treewidth of G is $w(0,\ell) - 1$. □

In the rest of this section we show that it is also easy to find a triangulation of G with minimum clique size. Notice that the algorithm described above can be easily adapted to return a triangulation T of the polygon $\mathcal{P}(Z)$ with minimum weight. Define the graph H(T) with the same vertex set as G as follows. Two vertices are adjacent in H(T) if there is a triangle such that the chords corresponding with the vertices intersect this triangle. Notice that G is a subgraph of H(T). We show that H(T) has a perfect elimination scheme. Consider a vertex z of $\mathcal{P}(Z)$ which is *not* incident with a diagonal of T. This vertex is incident with exactly one triangle Q. Consider a vertex x of which the corresponding chord intersects Q but no other triangle. Then the neighborhood of x, N(x), is a clique, hence x is simplicial. Notice also that the number of vertices in N(x) is the weight of Q minus one, showing that the number of vertices in the clique $\{x\} \cup N(x)$ is equal to the weight of Q. Remove x from the graph H(T) and the corresponding chord from the circle model. If there is no chord left in the circle model which intersects Q but no other triangle, then remove z from Z. Repeating this process gives a perfect elimination scheme, showing that H(T) is chordal. This also shows that the number of vertices in a maximum clique of H(T) is equal to the weight of the triangulation.

6 Acknowledgements

I like to thank M. de Berg, H. Bodlaender, A. Jacobs and H. Müller for valuable discussions.

References

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

[2] H.L. Bodlaender, Achromatic number is NP-complete for cographs and interval graphs, *Information Processing Letter* **31**, (1989), pp. 135–138.

[3] H. Bodlaender, A linear time algorithm for finding tree-decompositions of small treewidth, Technical report RUU-CS-92-12, Department of Computer Science, Utrecht University, Utrecht, The Netherlands, (1992). To appear in: STOC'93.

[4] H. Bodlaender, T. Kloks and D. Kratsch, Treewidth and pathwidth of permutation graphs, To appear in: *Proceedings of the 20th International Colloquium on Automata, Languages and Programming* (1993).

[5] H. Bodlaender and R.H. Möhring, The pathwidth and treewidth of cographs, In:*Proceedings 2nd Scandinavian Workshop on Algorithm Theory*, Springer Verlag, Lecture Notes in Computer Science 447, (1990), pp. 301–309. To appear in: *SIAM J. Discr. Math.*

[6] A. Brandstädt, Special graph classes — a survey, Schriftenreihe des Fachbereichs Mathematik, SM-DU-199 (1991) Universität Duisburg Gesamthochschule.

[7] G. A. Dirac, On rigid circuit graphs, *Abh. Math. Sem. Univ. Hamburg* **25**, (1961), pp. 71–76.

[8] M. Farber and M. Keil, Domination in permutation graphs, *J. Algorithms* **6**, (1985), pp. 309–321.

[9] C. P. Gabor, W. L. Hsu and K. J. Supowit, Recognizing circle graphs in polynomial time, *26th Annual IEEE Symposium on Foundations of Computer Science*, (1985).

[10] J. Gimbel, D. Kratsch and L. Stewart, On cocolourings and cochromatic numbers of graphs, to appear in: *Disc. Appl. Math.*

[11] M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

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

[13] T. Kloks, *Treewidth*, Ph. D. Thesis, Utrecht University, The Netherlands. To appear.

[14] T. Kloks and D. Kratsch, Treewidth of chordal bipartite graphs, *10th Annual Symposium on Theoretical Aspects of Computer Science*, Springer-Verlag, Lecture Notes in Computer Science 665, (1993), pp. 80–89.

[15] W. Naji, Reconnaissance des graphes de cordes, *Discrete Mathematics* **54**, (1985), pp. 329–337.

[16] R. Sundaram, K. Sher Singh and C. Pandu Rangan, Treewidth of circular arc graphs, To appear in: *SIAM J. Disc. Math.*

[17] K. J. Supowit, Finding a maximum planar subset of a set of nets in a channel, *IEEE Trans. Computer Aided Design* **6**, (1987), pp. 93–94.

[18] K. J. Supowit, Decomposing a set of points into chains, with applications to permutation and circle graphs, *Information Processing Letters* **21**, (1985), pp. 249–252.

[19] K. Wagner, Monotonic coverings of finite sets, *Journal of Information Processing and Cybernetics*, EIK, **20**, (1984), pp. 633–639.

[20] M. Yannakakis, Computing the minimum fill-in is NP-complete, *SIAM J. Alg. Disc. Meth.* **2**, (1981), pp. 77–79.