SIAM J. Comput.
Vol. 29, No. 2, pp. 378–386

WEAKLY TRIANGULATED COMPARABILITY GRAPHS*

ELAINE ESCHEN[†], RYAN B. HAYWARD[‡], JEREMY SPINRAD[§], AND R. SRITHARAN[¶]

Abstract. The class of weakly triangulated comparability graphs and their complements are generalizations of interval graphs and chordal comparability graphs. We show that problems on these classes of graphs can be solved efficiently by transforming them into problems on chordal bipartite graphs. We show that recognition and independent set on weakly triangulated comparability graphs can be solved in $O(n^2)$ time in this manner, and that the number of weakly triangulated comparability graphs is $2^{\Theta(n\log^2 n)}$. We also give algorithms to compute transitive closure and transitive reduction in $O(n^2\log\log n)$ time if the underlying undirected graph of the transitive closure is a weakly triangulated comparability graph.

Key words. weakly triangulated, comparability, partial orders, algorithm

AMS subject classifications. 05C70, 05C85, 06A06, 68Q25

PII. S0097539797322334

1. Introduction.

A variety of interesting graph classes correspond to requiring that the graph and/or its complement be a triangulated or a comparability graph. For example, permutation graphs are equal to graphs which are both comparability and cocomparability graphs, split graphs correspond to triangulated cotriangulated graphs, and interval graphs are exactly the chordal cocomparability graphs. These two properties are one of the themes which unify Golumbic's book on perfect graph classes [17]. We extend the study of the intersection of pairs of properties by generalizing from triangulated graphs to weakly triangulated graphs.

Triangulated graphs [28], also known as chordal graphs, are those graphs which contain no induced cycles of length greater than three. Weakly triangulated graphs allow induced four-cycles but do not allow any induced cycles of length greater than four in either the graph or its complement. We refer to an induced cycle of length five or more as a *hole*. We refer to the complement of a hole as an *antihole*. Weakly triangulated graphs properly contain chordal graphs and are a subclass of perfect graphs [18]. The fastest known recognition algorithm for weakly triangulated graphs takes $O(n^4)$ time [30]. The best algorithms for clique, independent set, chromatic number, and clique cover on weakly triangulated graphs also take $O(n^4)$ time [19, 3].

Comparability graphs [16], also known as transitively orientable graphs, are undirected graphs with the property that directions can be assigned to edges in such a way that whenever there is an edge from x to y and from y to z in the directed graph, there is also an edge from x to z. This assignment of direction to edges is called a

*Received by the editors June 2, 1997; accepted for publication (in revised form) June 25, 1998; published electronically October 5, 1999.
http://www.siam.org/journals/sicomp/29-2/32233.html

[†]Mathematics and Computer Science Department, Fisk University, 1000 17th Avenue North, Nashville, TN 37208 (eschen@dubois.fisk.edu). The work of this author was supported by the Kellogg Foundation grant "Building on Excellence in the Sciences" at Fisk University.

[‡]Department of Mathematics and Computer Science, University of Lethbridge, Lethbridge, AB T1K 3M4, Canada (hayward@cs.uleth.ca). The work of this author was supported by the Natural Science and Engineering Research Council of Canada.

[§]Department of Computer Science, Vanderbilt University, Nashville, TN 37235 (spin@vuse.vanderbilt.edu).

[¶]Department of Mathematics and Computer Science, Indiana State University, Terre Haute, IN 47809 (rst@berge.indstate.edu). The work of this author was supported by the University Research Committee, Indiana State University.

transitive orientation. Comparability graphs are also a subset of perfect graphs. The fastest known recognition algorithm for comparability graphs has two distinct stages. It is possible to find an orientation which is transitive if and only if the input is a comparability graph in $\mathrm{O}(n+m)$ time [25]. However, it is not known how to determine whether this orientation is actually transitive faster than performing a matrix multiplication, for which the current asymptotically fastest algorithm takes $\mathrm{O}(n^{2.376})$ time [7]. There are $\mathrm{O}(n+m)$ algorithms for clique and chromatic number [17, 25] on comparability graphs, if a transitive orientation is given. Maximum independent set on comparability graphs is essentially equivalent to bipartite matching [13], for which the best known algorithms have complexities of $\mathrm{O}(n^{1.5}\sqrt{m/\log n})$ [2] and $\mathrm{O}(\frac{n^{2.5}}{\log n})$ [11].

One of the most famous classes of intersection graphs is the class of interval graphs [4, 5]. A graph G is an interval graph if and only if both G is a chordal graph and the complement of G is a comparability graph [16]. Since weakly triangulated graphs are a natural generalization of triangulated graphs, we are interested to see whether the weakly triangulated cocomparability graphs are well behaved. Since weakly triangulated graphs are closed under complementation, this class is exactly the complement of weakly triangulated comparability graphs.

Weakly triangulated comparability graphs generalize the triangulated comparability graphs, studied in [23]. We note that both triangulated comparability graphs and triangulated cocomparability graphs can be recognized in linear time, which is faster than the algorithm obtained by separately testing whether G is triangulated and whether G is comparability/cocomparability. For weakly triangulated comparability and weakly triangulated cocomparability graphs, we give recognition algorithms which are faster than testing either whether G is weakly triangulated or whether G is a comparability/cocomparability graph.

Another class contained in the weakly triangulated comparability graphs is the class of permutation graphs [17]. This follows from the fact that cocomparability graphs cannot contain induced cycles of length greater than four [15], and permutation graphs are equal to graphs which are both comparability and cocomparability graphs.

Both the classes of comparability graphs and triangulated graphs contain $2^{\Theta(n^2)}$ graphs on n vertices, but the classes of triangulated comparability graphs and triangulated cocomparability graphs contain $2^{\Theta(n\log n)}$ graphs on n vertices. We show that the number of weakly triangulated comparability graphs on n vertices is $2^{\Theta(n\log^2 n)}$, which makes this class much smaller, for example, than the triangulated cotriangulated graph class, which has $2^{\Theta(n^2)}$ members.

The primary method used here to design efficient algorithms for weakly triangulated comparability graphs is to convert them into bipartite graphs and use the property that this transformation yields a chordal bipartite graph. The ideas are similar to those used in recognizing circular-arc graphs [10] and trapezoidal graphs [24] efficiently. Chordal bipartite graphs, studied in [21], are bipartite graphs in which every cycle length greater than four has a chord. The bipartite adjacency matrix of a graph is the 0/1 matrix formed by making one color class the rows, the other color class the columns, and placing a 1 at row x column y if and only if (x,y) is an edge of G. A 0/1 matrix has a Γ if there is some pair of rows $r_1 < r_2$ and columns $c_1 < c_2$ such that $(r_1,c_1) = (r_1,c_2) = (r_2,c_1) = 1$, while $(r_2,c_2) = 0$. The crucial theorem for this paper is that a graph is chordal bipartite if and only if the rows and columns of the bipartite adjacency matrix can be permuted to form a Γ-free matrix, and this ordering can be found in $\mathrm{O}(m\log n)$ or $\mathrm{O}(n^2)$ time [21, 27, 32].

2. Recognition. Let G be an arbitrary directed graph. Consider the undirected bipartite graph G' formed by making two copies x_1, x_2 of each vertex x and adding an edge from u_1 to v_2 if and only if there is an edge from u to v in G. We will call G' the *bipartite transformation* of G. Identical constructions have been made frequently; the most closely related work that uses this construction is due to Ford and Fulkerson [13], who use it for solving the independent set problem on partially ordered sets.

THEOREM 1. *If G is a transitively oriented graph, then the underlying undirected graph of G is weakly triangulated if and only if the bipartite transformation G' of G is chordal bipartite.*

Proof. First, we note that complements of comparability graphs cannot have induced cycles of length greater than four [15]. Thus, we do not have to be concerned with long antiholes in the underlying undirected graph of G.

Suppose that the underlying undirected graph of G is weakly triangulated. Consider a chordless cycle C of length at least six in G'. There cannot be a vertex x such that both x_1 and x_2 are in C; otherwise, there would be edges (y_1,x_2) and (x_1,u_2) in C, where $u_2 \neq y_2$ and (y_1,u_2) is not in C, and thus transitivity in G would imply that C has the chord (y_1,u_2). If x_1 and y_1 are both in C, then there cannot be an edge from x to y in G; otherwise, since G is transitively oriented, the neighborhood of x_1 would contain the neighborhood of y_1, and the vertices could not be part of the same chordless cycle. Similarly, if x_2 and y_2 are both in C, then there cannot be an edge from x to y in G. There cannot be any nonadjacent vertices u_1, v_2 in C such that there is an edge from v to u in G; otherwise, the edge (w_1,v_2) of C would imply an edge from w to u in G. Therefore, any chordless cycle C of G' corresponds to a chordless cycle of the same length in the underlying undirected graph of G; if the underlying graph of G is weakly triangulated, then G' must be chordal bipartite.

Suppose that G' is chordal bipartite. Let C be a chordless cycle in the underlying undirected graph of G. No vertex of C can have an edge directed to vertex u of C and an edge directed into it from vertex v of C, since v would have an edge to u by transitivity of the orientation. Let 1 be a vertex of C with edges directed to other vertices of C. If the cycle is $1, 2, \ldots, k$, then $1, 3, \ldots$ have edges directed outward with respect to other vertices of C, while $2, 4, \ldots$ have edges directed towards them. There will be a chordless cycle $1_1,\ 2_2,\ 3_1,\ 4_2, \ldots,\ (k-1)_1, k_2, 1_1$ in G', contradicting our assumption that G' is chordal bipartite. □

Theorem 1 immediately reduces the complexity of weakly triangulated comparability graph recognition to the time of comparability graph recognition and orientation plus the time of chordal bipartite graph recognition. Since comparability graphs can be recognized and transitively oriented in $O(n^{2.376})$ time using matrix multiplication [31], and chordal bipartite graphs can be recognized in $O(n^2)$ time [21, 27, 32], we have an $O(n^{2.376})$ algorithm for recognizing weakly triangulated comparability graphs. If fast matrix multiplication is not desired due to its complexity, we get an $O(n^3)$ recognition algorithm. These time complexities are an improvement over existing algorithms, since the best known algorithms for recognizing weakly triangulated graphs take $\Theta(n^4)$ time [30].

However, we can improve on these time bounds using properties of the comparability graph recognition algorithm. It is possible to find a transitive orientation of a comparability graph in $O(n+m)$ time [25]; the reason that this does not give a linear time algorithm for recognizing comparability graphs is that the algorithm will assign some direction to edges, even if no transitive orientation is possible. Therefore, we will use the following strategy for recognizing weakly triangulated comparability graphs.

We find an orientation of the graph which is transitive if and only if the graph has a transitive orientation. We determine whether the bipartite transformation of the directed graph is chordal bipartite; if not, then we know that G is either not a comparability graph or G is not weakly triangulated. We then use properties of chordal bipartite graphs, and use the bipartite transformation to verify that the orientation is transitive. If the orientation is transitive, we use Theorem 1 to answer that G is a weakly triangulated comparability graph.

We construct a Γ-free ordering of the bipartite transformation, which can be done in $\mathrm{O}(n^2)$ time [32]. Algorithms for constructing a Γ-free ordering work by using a technique called doubly lexical ordering of the matrix. Details of the procedure are not important to this paper, but if the neighborhood of a vertex x properly contains the neighborhood of vertex y, then doubly lexical ordering guarantees that the column or row corresponding to x comes after the column or row corresponding to y in the Γ-free ordering of the bipartite adjacency matrix.

THEOREM 2. *Suppose that the bipartite transformation of a directed graph G gives a chordal bipartite graph G'. We can determine whether G is transitive in* $\mathrm{O}(n^2)$ *time.*

Proof. For each pair of vertices x, y we can determine whether there is a transitivity violation $x \to y \to z$ in constant time. If x does not have an edge to y, clearly no such violation exists. So suppose there is an edge from x to y and x_1 comes before y_1 in the doubly lexical ordering of G'. There must be a vertex z such that $y \to z$ while x has no edge to z, or the neighborhood of y_1 would be properly contained in the neighborhood of x_1 (the containment is proper since x_1 has an edge to y_2 while y_1 does not), so we can immediately answer that G is not transitive. Now suppose that x_1 comes after y_1 in the ordering. Let z_2 be the first column of the Γ-free ordering of G' which has a 1 in row y_1. If x_1 does not have a 1 in this position, there is a transitivity violation $x \to y \to z$. If there is a 1 in this position, then there is no transitivity violation $x \to y \to u$, or the rows x_1, y_1 and columns z_2, u_2 would form a Γ. □

COROLLARY 3. *Weakly triangulated comparability graphs can be recognized in* $\mathrm{O}(n^2)$ *time.*

Proof. Given a graph G, we find an orientation of the edges which is transitive if and only if G is a comparability graph using the algorithm of [25, 31]. We construct the bipartite transformation G' of the directed version of G. By Theorem 1, if G is in the class, then G' must be a chordal bipartite graph; we test this in $\mathrm{O}(n^2)$ time using the algorithms in [21, 32]. If G' is chordal bipartite, we use Theorem 2 to test whether the orientation is actually transitive. If the orientation is transitive and G' is chordal bipartite, we use Theorem 1 to conclude that G is a weakly triangulated comparability graph. □

COROLLARY 4. *The number of weakly triangulated comparability graphs is* $2^{\Theta(n \log^2 n)}$.

Proof. Every chordal bipartite graph G is clearly a weakly triangulated comparability, since there are no induced cycles of length greater than four in G, and there cannot be any set of three vertices from a single color class in any hole of the complement. Any weakly triangulated comparability graph G can be reconstructed from the bipartite transformation of the transitive orientation of G, which is a chordal bipartite graph on $2n$ vertices. Therefore, the number of weakly triangulated comparability graphs on n vertices is some number between the number of chordal bipartite graphs on n vertices and the number of chordal bipartite graphs on $2n$ vertices. Since there are $2^{\Theta(n \log^2 n)}$ chordal bipartite graphs on n vertices [29], the result follows. □

3. Independent set, transitive closure, and transitive reduction. We know that there are polynomial algorithms for solving the clique, independent set, chromatic number, and clique cover problems on weakly triangulated comparability graphs, since such algorithms exist for these problems on both comparability graphs and weakly triangulated graphs. Clique and chromatic number can be solved in linear time on comparability graphs [17, 25], so we cannot hope to do better for these problems. Since these graphs are perfect, and the size of the largest independent set in G is equal to the clique cover number of G, we will consider only the independent set problem.

The best known algorithm for solving independent set on weakly triangulated graphs has time complexity $\Theta(n^4)$ [3, 19] and the best algorithm for computing the maximum independent set on comparability graphs has the same time complexity as finding a maximum matching in a bipartite graph [13], as described below. We show that this problem can be solved in $\mathrm{O}(n^2)$ time on weakly triangulated comparability graphs, as a simple consequence of Theorem 1. We note that it will be difficult to improve the time bound for independent set on general comparability graphs, since every bipartite graph is a comparability graph, and any improvement in the time complexity of computing maximum independent set on comparability graphs will improve the time complexity of computing the cardinality of a maximum matching in bipartite graphs.

The best algorithm for solving the independent set problem on comparability graphs is most easily phrased as solving the vertex cover problem on the class; to solve the maximum independent set problem, we simply take the vertices which are not part of the minimum vertex cover.

The algorithm for solving this problem takes a transitive orientation of the graph G, performs the bipartite transformation, and solves the vertex cover problem on the bipartite transformation G' using well known ideas for transforming bipartite vertex cover problem to bipartite matching problem. A vertex x is placed in the vertex cover of G if and only if either x_1 or x_2 is in the minimum vertex cover of G'; this will be the minimum vertex cover of G. The bottleneck step is solving the maximum matching problem on G'; the current best time bounds are $\mathrm{O}(n^{1.5}\sqrt{m/\log n})$ [2] and $\mathrm{O}(\frac{n^{2.5}}{\log n})$ [11].

For weakly triangulated comparability graphs, we make the same transformation, but instead of using a general matching algorithm we use the fact that G' is chordal bipartite, and use a simpler matching algorithm for G'. It is known that a maximum cardinality matching in a chordal bipartite graph can be found in linear time, given a Γ-free ordering of the bipartite adjacency matrix of the graph [6]. We include the algorithm here for the sake of completeness.

LEMMA 5. *The maximum cardinality matching problem on a chordal bipartite graph can be solved in* $\mathrm{O}(n^2)$ *time.*

Proof. We create a Γ-free ordering of the bipartite adjacency matrix for the chordal bipartite graph. We match the vertex corresponding to the first row with the vertex whose column corresponds to the first 1 entry in that row (if such a column exists), and delete that row and column. We then continue to the next row and repeat until all the rows and columns have been deleted. Consider any row r and column c which we choose for our matching. Suppose that we cannot choose to match these in a maximum cardinality matching that includes all previous matches. If r or c is unmatched in the optimal matching, then matching r with c cannot decrease the size of the matching. If r is matched with c_2 and c is matched with r_2, there must be an edge (r_2, c_2) or rows r, r_2 and c, c_2 induce a Γ. Therefore, we can match r with c and r_2 with c_2 and still get an optimum matching. □

Corollary 6. *The independent set problem can be solved in* $O(n^2)$ *time for weakly triangulated comparability graphs.*

Proof. Use Ford and Fulkerson's transformation [13] to reduce this to bipartite matching. Theorem 1 shows that this yields a chordal bipartite graph, and Lemma 5 lets us solve this problem in $O(n^2)$ time. □

Transitive closure is normally defined on a directed acyclic graph G; an edge from x to y is part of the transitive closure of G if there is a directed path of length ≥ 1 from x to y in G. The best algorithms known for general transitive closure involve matrix multiplication; relationships between matrix multiplication and transitive closure are examined in [12, 14, 26].

We will solve the transitive closure problem efficiently for any directed acyclic graph G with the property that the underlying undirected graph of the transitive closure is a weakly triangulated comparability graph. This problem can be solved in $O(n+m_t)$ time for chordal comparability graphs [23], where m_t is the number of edges of the transitive closure. We show that the problem can be solved in $O(n^2 \mathrm{loglog} n)$ time for weakly triangulated comparability graphs.

The algorithm most closely resembles the algorithm for transitive closure of two dimensional partial orders [22]. The fundamental idea is to use divide and conquer. The set of vertices is partitioned into two sets S_1, S_2 of size $\frac{n}{2}$ with the property that no edge goes from S_2 to S_1; the transitive closure is solved recursively within each set. The key task is to add edges which are implied by transitivity from S_1 to S_2; this is done by using the fact that the transitive closures of the two pieces can be transformed to chordal bipartite graphs using the bipartite transformation.

Lemma 7. *Suppose that G is a directed acyclic graph such that the underlying undirected graph is a weakly triangulated comparability graph. Assume that we are given the doubly lexical ordering of the bipartite transformation of G, and adjacency lists of each vertex ordered by column number in the ordering. Let x be a vertex which has edges to some vertices of G. We can find all edges out of x in the transitive closure of $\{x\} \cup G$ in* $O(n \mathrm{loglog} n)$ *time.*

Proof. Let $Y = y_1, y_2, \ldots, y_k$ be the vertices of G which have an edge from x. We place the vertices of Y in a priority queue, where the value associated with y_i is the column number of the first 1 in row y_i.

We repeatedly look at the vertex y_i with the smallest value in the priority queue; let z be the vertex with the column number that is currently associated with y_i. Suppose there is another vertex y_j on the queue with the same priority value as y_i. One of the vertices $\{y_i,y_j\}$ has smaller row number in the bipartite transformation; call the vertex with smaller row number y_a and the vertex with larger row number y_b. Since we are given a Γ-free ordering, for any vertex z' which comes after z in the ordering, if y_a has an edge to z', then y_b has an edge to z'. Therefore, we can eliminate y_a from our priority queue when looking for later edges implied by transitivity from x.

Therefore, the algorithm is as follows. Remove from the priority queue the vertex y_i with the smallest value. Let z be as above and suppose z's column number is k. Add an edge from x to z. Next examine the current smallest value on the queue. Suppose it is associated with the vertex y_j. If the value associated with y_j is also k, remove y_j from the priority queue and eliminate from further consideration whichever of y_i and y_j has the smaller row number. Repeat this step until the current smallest value in the queue is greater than k. The vertex with the largest row number among those with priority value k is then added back to the priority queue with associated value equal to the column index of the next 1 after column k in its row. Repeat this entire process until the priority queue is empty.

There are $\mathrm{O}(n)$ operations which correspond to adding an edge from x, and each priority queue operation which does not add an edge from x must delete one vertex from the priority queue. Therefore, the number of priority queue operations is $\mathrm{O}(n)$. If the priority queue is a heap, each operation takes $\mathrm{O}(\log n)$ time, and the total time spent is $\mathrm{O}(n\log n)$. However, each value in the queue is an integer in the range $1, \ldots, n$, which allows us to use van Emde Boas's data structure [33] with cost $\mathrm{O}(\log\log n)$ per operation. Therefore, all edges can be added in $\mathrm{O}(n\log\log n)$ time. $\square$

A similar procedure can be used to add a new vertex y with edges into it from vertices of G, and to add all edges implied by transitivity in $\mathrm{O}(n\log\log n)$ time.

THEOREM 8. *If the underlying undirected graph of the transitive closure of a directed acyclic graph G is a weakly triangulated comparability graph, then the transitive closure of G can be found in* $\mathrm{O}(n^2\log\log n)$ *time.*

Proof. Let T be a topological sort of G. Divide the vertices of G into two sets S_1, S_2, where S_1 consists of the first $\frac{n}{2}$ vertices in T and S_2 is the remainder. Recursively find the transitive closures C_1, C_2 of the subgraphs induced by S_1 and S_2. Construct the Γ-free matrix of the bipartite transformation of C_1 and C_2, and order the adjacency lists of each vertex of C_1 and C_2 so that vertices appear in increasing order of their columns in the Γ-free ordering.

For each vertex x in S_1, add all edges implied by transitivity of the form $x \rightarrow y_1 \rightarrow y_2$, where y_1 and y_2 are in S_2. By Lemma 7, this step takes $\mathrm{O}(n^2\log\log n)$ time overall. For each vertex y in S_2, add all edges implied by transitivity of the form $x_1 \rightarrow x_2 \rightarrow y$, $x_1, x_2 \in S_1$; this also takes $\mathrm{O}(n^2\log\log n)$ time.

At this point, we claim that we have the transitive closure of G. Consider any path from x to y in G. If x and y are both in S_1 or both in S_2, then the path must be entirely within S_1 or within S_2 as we divided vertices along a topological sort of G, and therefore, the edge from x to y will be added in the recursive step. Consider a path of G from x in S_1 to y in S_2. Let x_2 be the last vertex of S_1 on the path, and let y_2 be the first vertex of S_2 on the path. There must be an edge from x to x_2 in C_1, and an edge from y_2 to y in C_2. When we are adding edges implied by transitivity from x_2, we will add an edge from x_2 to y. When we are adding edges implied by transitivity into y, since the edge from x_2 to y has already been added, we will add an edge from x to y.

The running time of this algorithm is governed by the recurrence relation $\mathrm{T}(n) = 2\mathrm{T}(\frac{n}{2}) + \mathrm{O}(n^2\log\log n)$. $\mathrm{T}(n)$ can be shown to be $\mathrm{O}(n^2\log\log n)$; one automatic method for proving this can be found in [8]. $\square$

The transitive reduction of a directed acyclic graph G is formed by removing from G all edges $x \rightarrow z$ such that there exists a path of length greater than 1 from x to z in the transitive closure of G. The relationship between computing transitive closure and computing transitive reduction is discussed in [1]. We show that Lemma 7 can be used to find the transitive reduction of G in $\mathrm{O}(n^2\log\log n)$ time in the case that the underlying undirected graph of G's transitive closure is a weakly triangulated comparabilty graph.

THEOREM 9. *If the underlying undirected graph of the transitive closure of a directed acyclic graph G is a weakly triangulated comparability graph, then the transitive reduction of G can be found in* $\mathrm{O}(n^2\log\log n)$ *time.*

Proof. We first find the transitive closure of G and find a Γ-free ordering of the bipartite transformation of the transitive closure. Consider any vertex x. Let $y_1, y_2, \ldots, y_k$ be the vertices which have edges in from x in G. Consider adding a new vertex z which has edges to these same vertices. By Lemma 7, we can find all edges implied by transitivity from z in $\mathrm{O}(n\mathrm{loglog}n)$ time. An edge $x \to w$ of the transitive closure is implied by transitivity and therefore is not in the transitive reduction of G if and only if the edge $z \to w$ is found by the procedure of Lemma 7. Therefore, repeating the procedure for each vertex x in G, the transitive reduction of G can be found in $\mathrm{O}(n^2\mathrm{loglog}n)$ time. □

If the input graph is an arbitrary directed acyclic graph, Corollary 3 and Theorem 9 give us an $\mathrm{O}(n^2\mathrm{loglog}n)$-time algorithm which either finds the transitive reduction or declares that the underlying undirected graph of transitive closure of the input is not a weakly triangulated comparability graph. While computing the transitive closure of the input graph, we simply verify at each recursive step that the underlying undirected graph of the transitive closure of each subgraph is a weakly triangulated comparability graph.

4. Conclusions and open problems. We have shown that weakly triangulated comparability graphs can be dealt with more efficiently than either of the classes weakly triangulated or comparability individually, by transforming problems on these graphs to an associated chordal bipartite graph. This technique allows us to recognize and find maximum independent set in $\mathrm{O}(n^2)$ time for graphs in the class and to find transitive closure and reductions in $\mathrm{O}(n^2\mathrm{loglog}n)$ time if the transitive closure is in the class.

It would be interesting to find a geometric intersection model for the class of graphs or its complement. Extensions of the models for the subclasses of permutation graphs and interval graphs are possible, but the fact that there are $2^{\Theta(n\log^2 n)}$ weakly triangulated comparability graphs means that the models would have to be made considerably more complex.

Weakly triangulated cocomparability graphs are natural extensions of interval graphs, using the fact that a graph G is an interval graph if and only if G is a chordal cocomparability graph. Interval graphs are also equivalent to chordal asteroidal triple-free graphs [20], and cocomparability graphs are a proper subset of asteroidal triple-free graphs [9]. Thus, it also seems natural to consider weakly triangulated asteroidal triple-free graphs to see whether problems remain tractable on this larger class.

REFERENCES

[1] A. Aho, M. Garey, and J. Ullman, *The transitive reduction of a directed graph*, SIAM J. Comput., 1 (1972), pp. 131–137.

[2] H. Alt, N. Blum, K. Mehlhorn, and M. Paul, *Computing a maximum cardinality matching in a bipartite graph in time $O(n^{1.5}\sqrt{m/\log n})$*, Inform. Process. Lett., 37 (1991), pp. 237–240.

[3] S. Arikati and C. Rangan, *An efficient algorithm for finding a two-pair, and its applications*, Discrete Appl. Math., 31 (1991), pp. 71–74.

[4] S. Benzer, *On the topology of the genetic fine structure*, Proc. Nat. Acad. Sci. U.S.A., 45 (1959), pp. 1607–1620.

[5] K. S. Booth and G. S. Leuker, *Testing for the consecutive ones property, interval graphs, and planarity using PQ-tree algorithms*, J. Comput. System Sci., 13 (1976), pp. 335–379.

[6] M.-S. Chang, *Algorithms for maximum matching and mininum fill-in on chordal bipartite graphs*, in Algorithms and Computation, Lecture Notes in Comput. Sci. 1178, Springer-Verlag, Berlin, 1996, pp. 146–155.

[7] D. Coppersmith and S. Winograd, *Matrix multiplication via arithmetic progressions*, in Proceedings of the 19th Annual Symposium on the Theory of Computation, 1987, pp. 1–6.

[8] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 1989.

[9] D. Corneil, S. Olariu, and L. Stewart, *Asteroidal triple-free graphs*, in Graph-Theoretic Concepts in Computer Science, Lecture Notes in Comput. Science 790, Springer-Verlag, Berlin, 1994, pp. 211–224.

[10] E. Eschen and J. Spinrad, *An $O(n^2)$ algorithm for circular-arc graph recognition*, in Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, 1993, pp. 128–137.

[11] T. Feder and R. Motwani, *Clique partitions, graph compression and speeding up algorithms*, Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, 1991, pp. 123–133.

[12] M. J. Fischer and A. R. Meyer, *Boolean matrix multiplication and transitive closure*, in Conference Record, Twelfth Annual Symposium on Switching and Automata Theory, East Lansing, MI, 1971, pp. 129–131.

[13] L. Ford and D. Fulkerson, *Flows in Networks*, Princeton University Press, Princeton, NJ, 1962.

[14] M. E. Furman, *Applications of a method of fast multiplication of matrices in the problem of finding the transitive closure of a graph*, Dokl. Akad. Nauk SSSR, 11 (1970), p. 1252.

[15] T. Gallai, *Transitiv orientbare graphen*, Acta Math. Acad. Sci. Hung. Tom., 18 (1967), pp. 25–66.

[16] P. C. Gilmore and A. J. Hoffman, *A characterization of comparability graphs and interval graphs*, Canad. J. Math., 16 (1964), pp. 539–548.

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

[18] R. Hayward, *Weakly Triangulated Graphs*, J. Combin. Theory Ser. B, 39 (1985), pp. 200–209.

[19] R. B. Hayward, C. T. Hoàng, and F. Maffray, *Optimizing weakly triangulated graphs*, Graphs Combin., 5 (1989), pp. 339–349; erratum in 6 (1990), pp. 33–35.

[20] C. G. Lekkerkerker and J. Boland, *Representation of a finite graph by a set of intervals on the line*, Fund. Math., 51, pp. 45–64.

[21] A. Lubiw, *Doubly lexical orderings of matrices*, SIAM J. Comput., 16 (1987), pp. 854–879.

[22] T.-H. Ma and J. Spinrad, *Transitive closure for restricted classes of partial orders*, Order, 8 (1991), pp. 175–183.

[23] T.-H. Ma and J. Spinrad, *Cycle-free partial orders and chordal comparability graphs*, Order, 8 (1991), pp. 49–61.

[24] T.-H. Ma and J. Spinrad, *on the 2-chain subgraph cover and related problems*, J. Algorithms, (1994), pp. 251–268.

[25] R. McConnell and J. Spinrad, *Linear-time transitive orientation*, in Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, 1997, pp. 19–25.

[26] I. Munro, *Efficient Determination of the Strongly Connected Components and the Transitive Closure of a Graph*, unpublished manuscript, University of Toronto, 1971.

[27] R. Paige and R. E. Tarjan, *Three partition refinement algorithms*, SIAM J. Comput., 16 (1987), pp. 973–989.

[28] D. J. Rose, R. E. Tarjan, and G. S. Leuker, *Algorithmic aspects of vertex elimination on graphs*, SIAM J. Comput., 5 (1976), pp. 266–283.

[29] J. Spinrad, *Nonredundant ones and chordal bipartite graphs*, SIAM J. Discrete Math., 8 (1995), pp. 251–257.

[30] J. Spinrad and R. Sritharan, *Algorithms for weakly triangulated graphs*, Discrete Appl. Math., 19 (1995), pp. 181–191.

[31] J. P. Spinrad, *On comparability and permutation graphs*, SIAM J. Comput., 14 (1987), pp. 658–670.

[32] J. P. Spinrad, *Doubly lexical ordering of dense 0-1 matrices*, Inform. Process. Lett., 45 (1993), pp. 229–235.

[33] P. Van Emde Boas, *Preserving order in a forest in less than logarithmic time and linear space*, Inform. Process. Lett., 6 (1977), pp. 80–82.