Efficient Parallel Algorithms for Recognizing Weakly Triangulated Graphs

Stavros D. Nikolopoulos

05-97

Technical Report No. 05-97/1997

Department of Computer Science
University of Ioannina
45 110 Ioannina, Greece

Efficient Parallel Algorithms for Recognizing Weakly Triangulated Graphs

STAVROS D. NIKOLOPOULOS

Department of Computer Science, University of Ioannina,
P.O. Box 1186, GR-45110 Ioannina, Greece
stavros@cs.uoi.gr

Abstract - We present an efficient parallel algorithm for detecting chordless cycles of length $k \geq 5$ in undirected graphs, which runs in $O(\log n)$ time using $O(n^{4.376})$ processors on a CRCW PRAM model of computation. Our results directly imply that weakly triangulated graphs can be recognized in $O(\log n)$ time using $O(n^{4.376})$ processors and, thus, improve in performance upon the best-known parallel algorithm for recognizing weakly triangulated graphs [5], which runs in $O(\log n)$ time using $O(n^5)$ processors on a CRCW PRAM. Moreover, we present an efficient parallel algorithm for recognizing triangulated graphs, runing in $O(\log n)$ time using $O(n^{3.376})$ processors on a CRCW PRAM model of computation.

Keywords: Parallel algorithms, triangulated graphs, weakly triangulated graphs, chordless cycles, graph partition, CRCW PRAM, complexity.

1. Introduction

A cycle C = (v_0, v_1, v_2, ..., v_l, v_0) in an undirected graph G = (V, E) is called *simple cycle* if $v_i \neq v_j$ for $i \neq j$. A simple cycle is *chordless* if $(v_i, v_j) \notin$ E for i and j differing by more than 1 *mod* l+1. An undirected graph G is said to be *triangulated* (chordal) if it has no chordless cycle of length greater than or equal to 4 (see Golumbic [10]), while G is said to be *weakly triangulated* if both G and the complement of G have no chordless cycle of length greater than or equal to 5 (see Hayward [11]). Triangulated graphs arise in the study of Gaussian elimination on sparse symmetric matrices [16, 17, 18, 22], in the study of acyclic relational schemes [3], and are related to and useful for many location problems [10, 12].

Our objective is to design efficient parallel algorithms for detecting chordless cycles of length greater than or equal to 4 and 5, which in turn lead to efficient parallel algorithms for recognizing triangulated and weakly triangulated graphs.

Many recognition algorithms have been developed for triangulated graphs, operating in a sequential and/or parallel process environment. Fulkerson and Gross [9] suggested an iterative procedure to recognize triangulated graphs and pointed out properties for some other objects of such graphs. Edenbrandt [7] proposed a parallel recognition algorithm which is running in

$O(\log n)$ time with $O(n^5)$ processors on a CRCW PRAM or in $O(\log^2 n)$ time with $O(n^5)$ processors on a CREW PRAM. Chandrasekharan and Iyengar [4] proposed a parallel algorithm for recognizing triangulated graphs which can be executed in $O(\log n)$ time with $O(n^4)$ processors on a CRCW PRAM. Naor, Naor and Schäffer [15] proposed a parallel recognition algorithm which runs in time $O(\log^2 n)$ by using $O(n^4)$ processors on a CREW PRAM. They also proposed parallel algorithms for some other problems (e.g. maximal cliques) on triangulated graphs which runs in $O(\log^3 n)$ time using $O(n^4)$ processors or in $O(\log^2 n)$ time using $O(n^5)$ processors on the same type of computational model. Klein [14] has announced efficient parallel algorithms for several problems on triangulated graphs, among which algorithms for the recognition problem, which run in time $O(\log^2 n)$ using $O(m+n)$ processors on a CRCW PRAM, where m is the number of edges in the graph. After the Klein's publication, Ho and Lee [12] formulated an algorithm which computes a clique tree in $O(\log n)$ time with $O(n^3)$ processors on a CRCW PRAM. Subsequently these authors [13] formulated an algorithm which, given a clique tree of a graph, computes a perfect elimination scheme in $O(\log n)$ time with $O(n^2)$ processors in the same type of computational model. This implies that a triangulated graph can be recognized in $O(\log n)$ time with $O(n^3)$ processors on a CRCW PRAM. Moreover, these authors [12] proposed an algorithm which computes some other objects of a chordal graph in $O(\log n)$ time on a CRCW PRAM or in $O(\log^2 n)$ time on a CREW PRAM using $O(n^3)$ processors. Table 2 shows existing results of parallel solutions to the triangulated graph recognition problem.

Table 1. Parallel algorithms for recognizing triangulated graphs.

Authors	Time	Processors	Model
Edenbrandt [7]	$O(\log n)$	$m\,n^3$	CRCW
Chandrasekharan and Iyengar [4]	$O(\log n)$	n^4	CRCW
Naor, Naor and Schäffer [15]	$O(\log^2 n)$	$m\,n^2$	CRCW
Klein [14]	$O(\log^2 n)$	$m+n$	CRCW
Ho and Lee [13]	$O(\log n)$	n^3	CRCW

The problem of recognizing weakly triangulated graphs have been extensively studied, mainly in the context of finding chordless cycles of length $k \geq 5$ [20]. Hayward [11] proposed an $O(m\,n^3)$-time sequential algorithm for detecting a chordless cycle of length greater than or equal to 5, which leads to a recognition algorithm for weakly triangulated graphs in $O(n^5)$ time. Hayward's results imply a parallel recognition algorithm for weakly triangulated graphs running in $O(\log n)$ time with $O(n^5)$ processors on a CRCW PRAM. The work of Sritharan and Spinrad [21] provides a sequential algorithm for recognizing weakly triangulated graphs in $O(m\,n^2)$

time. Unfortunately, this algorithm does not seem to be amenable to parallelization. Recently, Chandrasekharan *et. al.* [5] presented a parallel algorithm for obtaining a chordless cycle of length greater than or equal to $k \geq 4$ in a graph in $O(m^2n^{k-4})$ time sequentially and in $O(\log n)$ time using $O(m^2n^{k-4})$ processors in parallel on a CRCW PRAM, whenever such a cycle exists. By setting $k = 4$ we see that a chordless cycle of length greater than or equal to 4 can be found in $O(\log n)$ time using $O(m^2)$ processors, while by setting $k = 5$ a chordless cycle of length greater than or equal to 5 can be found in $O(\log n)$ time using $O(m^2n)$ processors. These results lead to parallel algorithms for recognizing triangulated and weakly triangulated graphs running in $O(\log n)$ time on a CRCW PRAM using $O(n^4)$ and $O(n^5)$ processors, respectively. Table 2 shows some existing results on recognizing weakly triangulated graphs in parallel.

Table 2. Parallel algorithms for recognizing weakly triangulated graphs.

Authors	Time	Processors	Model
Hayward [11]	$O(\log n)$	n^5	CRCW
Chandrasekharan *et. al.* [5]	$O(\log n)$	n^5	CRCW

In this paper we present efficient parallel algorithms for recognising weakly triangulated graphs. Our technique is based on the notion of partitioning the vertex set V of a graph G, with respect to a vertex v, into a set of (mutually disjoint) adjacency-level sets $N_0(v)$, $N_1(v)$, ..., $N_L(v)$, $0 \leq L < n$. Specifically, we proposed a parallel algorithm for detecting a chordless cycle of length $k \geq 5$, in $O(\log n)$ time using $O(n^{4.376})$ processors on a CRCW PRAM computational model, if such a cycle exists. These results directly imply that weakly triangulated graphs can be recognized in $O(\log n)$ time using $O(n^{4.376})$ processors on a CRCW PRAM. Moreover, based on the same technique we can easily show that triangulated graphs are recognized in $O(\log n)$ time using $O(n^{3.376})$ processors on a CRCW PRAM.

The main result of this paper for recognizing weakly triangulated graphs improves in performance upon the best-known parallel algorithm which recognizes weakly triangulated graphs in $O(\log n)$ time using $O(n^5)$ processors on a CRCW PRAM model of computation [5].

2. Graph Partition

Given a graph G=(V,E) and a vertex $v \in$ V, we define a partition $\mathcal{L}(G, v)$ of the vertex set V (we shall frequently use the term *partition of the graph* G), with respect to the vertex v as follows:

$$\mathcal{L}(G, v) = \{ N_i(v) \mid v \in V, 0 \leq i \leq L_v, 1 \leq L_v < |V| \}$$

where $N_i(v)$, $0 \leq i \leq L_v$, are the *adjacency-level sets*, or simply the *adjacency-levels*, and L_v is the

length of the partition $\mathcal{L}(G, v)$ [16]. The adjacency-level sets of the partition $\mathcal{L}(G, v)$ of the graph G, are defined as follows:

$$N_i(v) = \{u \mid d(v, u) = i,\ 0 \le i < n\}$$

where $d(v, u)$ denotes the *distance* between vertices v and u in G and $n = |V|$. We point out that $d(v, u) \ge 0$, and $d(v, u) = 0$ when $v = w$, for every $v, w \in V$. In the case where G is a disconnected graph, $d(v, u) = \infty$ when v and w do not belong to the same connected component. Obviously, $L_v = max\ \{d(v, u) \mid u \in V\}$, $N_0(v) = \{v\}$ and $N_1(v) = \{u \mid (v, u) \in E\}$.

We can extend the notion of the adjacency-level sets so that for any set $S \subseteq V$ we define $N_0(S) = S$ and $N_i(S) = \{u \in S \mid d(v, u) = i$ and $v \in S,\ 1 \le i < n\}$. In fig. 1 we illustrate the adjacency-level sets $N_0(x, y)$, $N_1(x, y)$, ..., $N_L(x, y)$ of a graph, where $N_0(x, y) = \{x, y\}$, $N_1(x, y) = \{x_1, x_2, x_3, y_1, y_2\}$ and so on. Throughout the paper, we shall use the notation $N_i(x, y)$ instead of $N_i(\{x, y\})$, $0 \le i < n$.

The adjacency-level sets $N_i(v)$, $0 \le i \le L_v$, of partition $\mathcal{L}(G, v)$, can easily be computed recursively as follows: $N_i(v) = \{u \mid (x, u) \in E$ and $x \in N_{i-1}(v)\} - \{N_{i-1}(v) \cup N_{i-2}(v)\}$, $2 \le i \le L_v < n$. Using a CRCW PRAM, the adjacency-level sets can be computed in $O(n)$ time with $O(n^2)$ processors. Moreover, these sets can also be computed by considering first the distance matrix of the graph G and then extracting all set information that is necessary. This computation can be done in $O(\log n)$ time by using $O(n^{\beta}+D_G)$ processors, where $\beta = 2.376$ and D_G is the output size of the partitions of the graph, see [6]. On the other hand, it might well be possible to get nearly optimal complexity, i.e., $\Theta(D_G)$ processors, with the techniques given, e.g., in [1].

3. The Main Results

In this section we present a parallel algorithm for detecting a chordless cycle of length $k \ge 5$ in an undirected graph $G = (V, E)$ whenever such a cycle exists. Towards this algorithmic process, for each pair of vertices $x, y \in V$ such that $(x, y) \in E$ we define two vertex sets as follows:

$\boldsymbol{FV_{\{x, y\}}}$: it contains all the vertices z that are adjacent to both vertices x and y, i.e.,

$$FV_{\{x, y\}} = \{\ z \in V \mid (x, z) \in E \text{ and } (y, z) \in E\}$$

$\boldsymbol{AV^x_{\{x, y\}}}$ (resp. $\boldsymbol{AV^y_{\{x, y\}}}$): it contains all the vertices z that are adjacent to x (resp. y) and are not adjacent to y (resp. x), i.e.,

$$AV^x_{\{x, y\}} = \{\ z \in V \mid (x, z) \in E \text{ and } (y, z) \notin E\}$$

Thus, for the pair of vertices x, y of the graph of Fig. 1, the above defined vertex sets are $FV_{\{x, y\}} = \{x_2\}$, $AV^x_{\{x, y\}} = \{x_1, x_3\}$ and $AV^y_{\{x, y\}} = \{y_1, y_2\}$. Note that, all the elements of both sets $AV^x_{\{x, y\}}$ and $AV^y_{\{x, y\}}$ belong to the set $N_1(x, y)$.

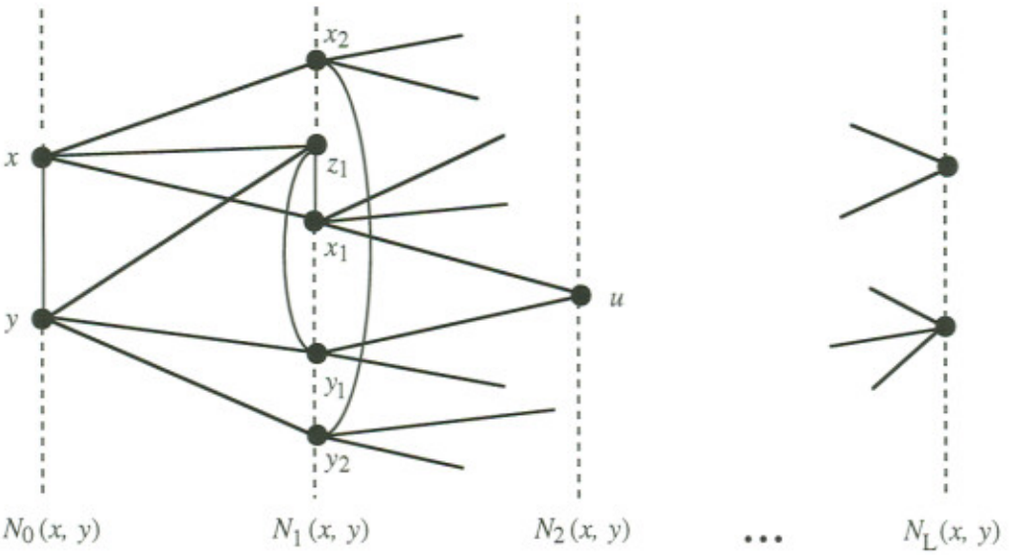

Fig. 1: The adjacency-level sets $N_0(x, y)$, $N_1(x, y)$, ..., $N_L(x, y)$ of a graph G = (V, E), with respect to the vertex set $\{x, y\}$. Here, $(x, y) \in$ E.

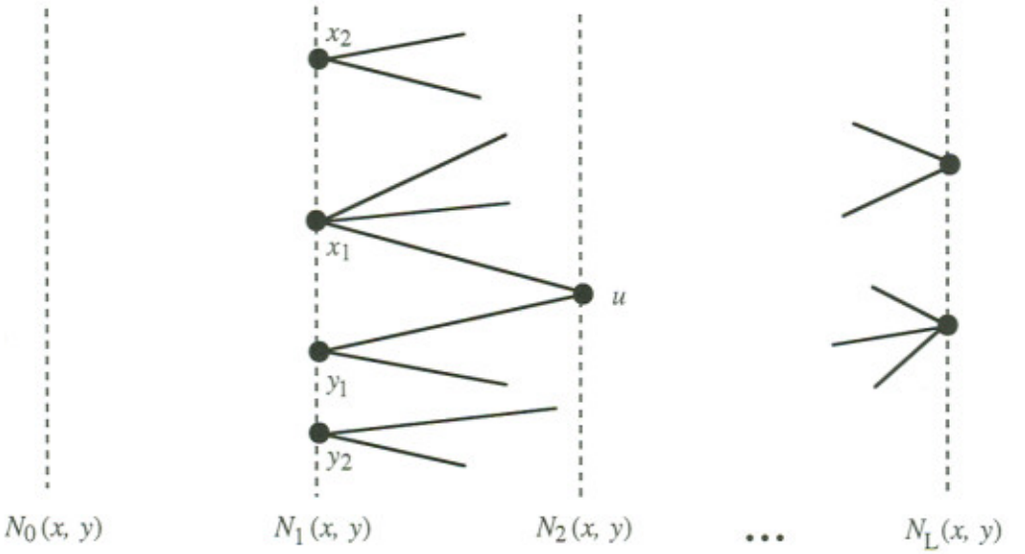

Fig. 2: The undirected graph $G'_{xy} = (V'_{xy}, E'_{xy})$ of the graph G = (V, E) of the fig. 1.

Given an edge $(x, y) \in$ E and the adjacency-level sets $N_0(x, y)$, $N_1(x, y)$, ..., $N_L(x, y)$ of the partition $\hat{L}(G, S)$, where $S = \{x, y\}$, we define an undirected graph G'_{xy} as follows:

$V'_{xy} = V - \{x, y\} - FV_{\{x, y\}}$, and

$E'_{xy} = E - \{(w, w') \in E \mid w, w' \in N_1(x, y)\}$

where $FV_{\{x, y\}}$, is the vertex set containing all the vertices that are adjacent to both vertices x and y (see fig. 2).

Having defined the undirected graph $G'_{xy} = (V'_{xy}, E'_{xy})$ of a graph $G = (V, E)$, where $(x, y) \in E$, let us now define a directed graph $G''_{xy} = (V''_{xy}, E''_{xy})$ which will be the key graph in our recognition algorithm. So, $G''_{xy} = (V''_{xy}, E''_{xy})$ is a directed graph such that:

(i) $x \in V''_{xy}$ iff $x \in V'_{xy}$; that is $V''_{xy} = V'_{xy}$.

(ii) $\langle x, u\rangle \in E''_{xy}$ if $(x, u) \in E'_{xy}$ and $x \in AV^x_{\{x, y\}}$,
$\langle u, y\rangle \in E''_{xy}$ if $(u, y) \in E'_{xy}$ and $y \in AV^y_{\{x, y\}}$.
$\langle u, u'\rangle \in E''_{xy}$ and $\langle u', u\rangle \in E''_{xy}$ if $(u, u') \in E'_{xy}$ and $u, u' \notin AV^y_{\{x, y\}} \cup AV^y_{\{x, y\}}$.

The directed graph $G''_{xy} = (V''_{xy}, E''_{xy})$ of the graph $G'_{xy} = (V'_{xy}, E'_{xy})$ of fig. 2, is presented in fig. 3.

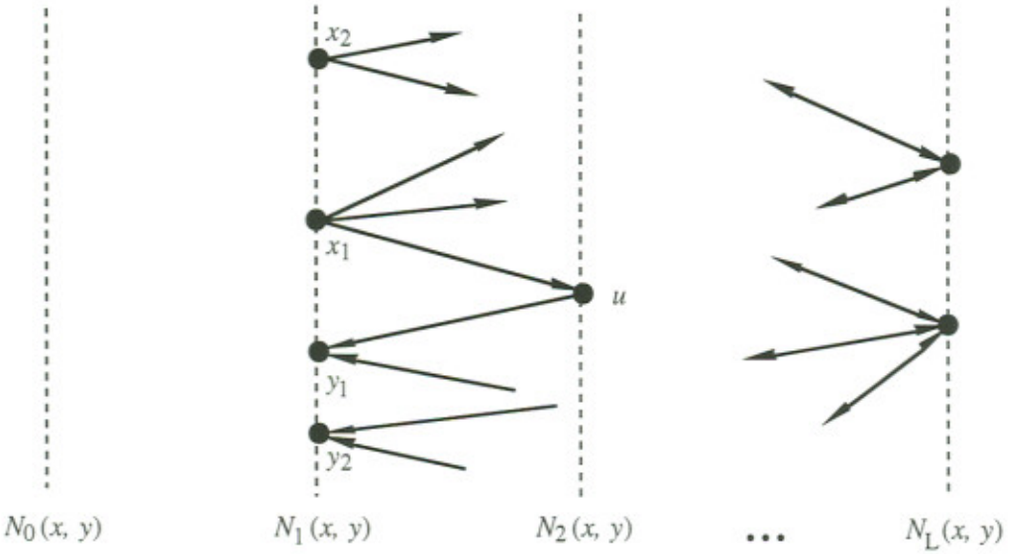

Fig. 3: The directed graph $G''_{xy} = (V''_{xy}, E''_{xy})$ of the graph $G'_{xy} = (V'_{xy}, E'_{xy})$ of the fig. 2.

The following results provide some characterizations of the adjacency-level sets and the structure of a chordless cycle of an undirected graph, leading to an efficient parallel algorithms for detecting C_k, $k \geq 5$. We state the following lemma which is the basis of the algorithm.

Lemma 1. A graph $G = (V, E)$ contains a chordless cycle C_k, $k \geq 5$, if and only if there exists an edge $(x, y) \in E$ and vertices $x_1, y_1 \in V$, such that:

(i) $x_1 \in AV^x_{\{x, y\}}$ and $y_1 \in AV^y_{\{x, y\}}$,

(ii) $(x_1, y_1) \notin E$ and,

(iii) there exists a directed path from x_1 to y_1 in $G''_{xy} = (V''_{xy}, E''_{xy})$.

Proof. (Sufficiency) Consider an edge $(x, y) \in E$ and suppose there exist vertices $x_1 \in AV^x_{\{x, y\}}$ and $y_1 \in AV^y_{\{x, y\}}$ satisfying the required conditions (see fig. 1; note that any pair of vertices in $N_1(x, y)$, except for $\{x_1, y_1\}$, may be adjacent). Since $(x_1, y_1) \notin E$ and $x_1 \rightarrow y_1$ in graph G''_{xy}, there is a chordless path of length at least equal to 2 in G'_{xy} starting at x_1 and ending at y_1, i.e., $P_{k'} = (x_1, ..., u, ..., y_1)$, $k' \geq 3$. Moreover, the path (x_1, x, y, y_1) in G is a chordless path of length

3; by definition, $(x, y) \in E$ whereas $(x_1, y) \notin E$, $(y_1, x) \notin E$. This implies that there exists a cycle of length at least equal to 5 in G, i.e., $C_k = (x, x_1, ..., u, ..., y_1, y, x)$, $k \geq 5$. Since $G'(N_1(x, y))$ is a mK_1 graph, $m \geq 2$, all the vertices (except x_1 and y_1) of the chordless path $P_{k'} = (x_1, ..., u, ..., y_1)$ belong to the set $N_2(x, y) \cup ... \cup N_L(x, y)$. Obviously then, $(x, u) \notin E$ and $(y, u) \notin E$. Thus, the cycle $C_k = (x, x_1, ..., u, ..., y_1, y, x)$ is a chordless cycle of length $k \geq 5$.

(Necessity) Suppose there exists a chordless cycle $C_k = (v_1, v_2, v_3, ..., v_{k-1}, v_k, v_1)$ in G of length $k \geq 5$. We set $v_1 = x$ and $v_k = y$ and we consider the vertex pair $\{x, y\}$ and the partition $N_0(x, y), N_1(x, y), ..., N_L(x, y)$. Obviously, vertices $v_2 = x_1$ and $v_{k-1} = y_1$ satisfy the required condition, since the vertices $v_3, v_4, ..., v_{k-2}$ belong to $N_2(x, y) \cup N_3(x, y) \cup ... \cup N_L(x, y)$ and $x_1 = v_2, v_3, ..., v_{k-1} = y_1$ is a chordless path in G' (actually, $x_1 = v_2, v_3, ..., v_{k-1} = y_1$ is a directed path in G''). □

We are now in a position to formulate an algorithm for the parallel detection of a chordless cycle of length greater than or equal to 5 in an undirected graph. The algorithm is based on the results provided by Lemma 1. The algorithm has input the adjacency matrix of the given graph G and operates as follows:

Algorithm `5_Chordless_Cycle`

begin

1. for every pair $x, y \in V$, do in parallel
   if $(x, y) \in E$ then
   1.1 compute the vertex sets $FV_{\{x, y\}}$, $AV^x_{\{x, y\}}$ and $AV^y_{\{x, y\}}$;
   1.2 compute the adjacency-level set $N_1(x, y)$;
   1.3 compute the graph $G'_{xy} = (V'_{xy}, E'_{xy})$, having the following vertex and edge sets:
   $V'_{xy} = V - \{x, y\} - FV_{\{x, y\}}$;
   $E'_{xy} = E - \{(w, w') \in E \mid w, w' \in N_1(x, y)\}$;
   and then the directed graph $G''_{xy} = (V''_{xy}, E''_{xy})$;
   1.4 compute the distance matrix of the graph $G''_{xy} = (V''_{xy}, E''_{xy})$;
   1.5 for every $\{x_1, y_1\}$, such that $x_1 \in AV^x_{\{x, y\}}$ and $y_1 \in AV^y_{\{x, y\}}$ do in parallel
   if there exists a directed path from x_1 to y_1 in $G''_{xy} = (V''_{xy}, E''_{xy})$
   and $(x_1, y_1) \notin E$, then $Found_{\{x, y\}} \leftarrow$ true;
2. if there is a vertex pair $\{x, y\}$ such that $Found_{\{x, y\}}$ = true, then
   G contains a C_k, $k \geq 5$;

end.

The correctness of the parallel algorithm `5_Chordless_Cycle` is established through the Lemma 1. Next, we analyze the computational complexity of the algorithm. We shall obtain its overall complexity by computing the complexity of each step separately. As a model of parallel computation, we use a Concurrent-Read, Concurrent-Write Parallel RAM (CRCW PRAM) [2, 8].

The complexity of the parallel algorithm is analyzed as follows: *Step 1*. This step consists of five substeps. *Substep 1*: The computation of the vertex sets $FV_{\{x, y\}}$, $AV^x_{\{x, y\}}$ and $AV^y_{\{x, y\}}$ can be completed in $O(1)$ time using $O(n^2)$ processors or in $O(\log n)$ time using $O(n^2 / \log n)$ processors. *Substep 2*: The adjacency-level set $N_2(x, y)$ can be computed $O(1)$ time with $O(n)$ processors. *Substep 3*: Given the adjacency matrix of the graph G and the vertex sets $FV_{\{x, y\}}$ and $N_2(x, y)$, the adjacency matrix of the graph G' can be computed $O(1)$ time using $O(n^2)$ processors. Moreover, the adjacency matrix of the graph G'' can be computed within the same time-processor bounds. *Substep 4*: The distance matrix of a graph can be found in $O(\log n)$ time using $n^{2.376}$ processors on a CRCW PRAM by Coppersmith and Winograd's technique [6]. *Substep 5*: It is well-known that once the distance matrix of a graph are computed, we can answer quires of the form "is there a directed path from u to v?" in $O(1)$ sequential time. Here, n^2 pair of vertices $\{x_1, y_1\}$ are tested for $x_1 \rightarrow y_1$. Thus, this substep can be executed in $O(1)$ time when $O(n^2)$ processors are available. In total, step 1 is executed in $O(\log n)$ time with $O(mn^2)$ processors. *Step 2*. It is easy to see that this step is executed in $O(1)$ time using $O(m)$ processors.

Taking into consideration the time-processor complexity of each step of the algorithm, we can obtain the overall computational complexity of the algorithm. Thus, we present the following result.

Lemma 2. Given an undirected graph G=(V,E), algorithm 5_Chordless_Cycle correctly detects a chordless cycle of length $k \geq 5$, in $O(\log n)$ time using $O(mn^{2.376})$ processors on a CRCW PRAM model of computation, if such a cycle exists.

The complement co-G of a graph G can be computed in $O(\log n)$ time using $O(n^2/\log n)$ processors on a CRCW PRAM computational model. Thus, we obtain the following Theorem.

Theorem 1. Weakly triangulated graphs can be recognized in $O(\log n)$ time using $O(n^{4.376})$ processors on a CRCW PRAM model of computation.

4. Recognizing Triangulated Graphs

In this section we present a parallel algorithm for recognizing triangulated graphs. Specifically, we present a parallel algorithm for detecting chordless cycles of length $k \geq 4$ in an undirected graph. The result of this section can be immediately derived from Lemma 1, if we consider the partition $\mathcal{L}(G, v)$ instead of $\mathcal{L}(G, \{x, y\})$, where G = (V, E) is an undirected graph and $v, x, y \in$ V. Thus, we state the following lemma which is the basis of the algorithm.

Lemma 3. A graph G = (V, E) contains a chordless cycle C_k, $k \geq 4$, if and only if there exists a vertex $v \in$ V and vertices $x_1, y_1 \in$ V, such that:

(i) $x_1, y_1 \in N_1(v)$,

(ii) $(x_1, y_1) \notin$ E and,

(iii) there exists a directed path from x_1 to y_1 in $G''_{xy} = (V''_{xy}, E''_{xy})$.

Proof. Immediately from Lemma 1 by partitioning the graph G with respect to v and setting $FV_{\{x, y\}} = \varnothing$. □

Based on the above results, we obtain the following parallel algorithm for detecting a chordless cycle C_k in an undirected graph, where $k \geq 4$.

Algorithm `4_Chordless_Cycle`

begin

1. for every vertex $v \in V$, do in parallel
   - 1.1 compute the adjacency-level set $N_1(v)$;
   - 1.2 compute the graph $G'_{xy} = (V'_{xy}, E'_{xy})$, having the following vertex and edge sets:
     $V'_{xy} = V - \{x, y\} - FV_{\{x, y\}}$;
     $E'_{xy} = E - \{(w, w') \in E \mid w, w' \in N_1(x, y)\}$;
     and then the directed graph $G''_{xy} = (V''_{xy}, E''_{xy})$;
   - 1.3 compute the distance matrix of the graph $G''_{xy} = (V''_{xy}, E''_{xy})$;
   - 1.4 for every pair $\{x_1, y_1\}$, such that $x_1, y_1 \in N_1(v)$ do in parallel
     if there exists a directed path from x_1 to y_1 in $G''_{xy} = (V''_{xy}, E''_{xy})$
     and $(x_1, y_1) \notin E$, then $Found_{\{v\}} \leftarrow$ true;
2. if there is a vertex v such that $Found_{\{v\}}$ = true, then
   G contains a C_k, $k \geq 4$;

end.

Having proved the correctness of the algorithm `5_Chordless_Cycle` (see Lemma 2), it is easy to show that the algorithm `4_Chordless_Cyclelet` correctly detects a chordless cycle of length $k \geq 4$. As far as its computational complexity is concerned, it is also easy to show that the step 1 is executed in $O(\log n)$ time using $O(n n^{2.376})$ processors, while step 2 is executed in $O(1)$ time using $O(n)$ processors. Again, we use a CRCW PRAM as a model of parallel computation. Thus, we have the following result.

Theorem 2. Triangulated graphs can be recognized in $O(\log n)$ time using $O(n^{3.376})$ processors on a CRCW PRAM model of computation.

5. Concluding Remarks

We have presented efficient CRCW $O(\log n)$-time parallel algorithms for detecting chordless cycles of length $k \geq 5$, in an undirected graph, using $O(n^{4.376})$ processors, respectively. These results directly imply that weakly triangulated graphs can be recognised in $O(\log n)$ time using $O(n^{4.376})$ processors on a CRCW PRAM. Moreover, we have shown that triangulated graphs can be recognised in $O(\log n)$ time using $O(n^{3.376})$ processors on a CRCW PRAM.

The results of this paper improve in performance upon the best-known parallel algorithm for recognizing weakly triangulated graph [5], which runs in $O(\log n)$ time using $O(n^5)$ processors on a CRCW PRAM model of computation. Moreover, the efficiency of the proposed weakly triangulated parallel recognition algorithm is approximately $1/\log n$, since the best bound for the sequential case is $O(n^4)$ due to work of Sritharan and Spinrad [21].

References

[1] B. Awerbuch and Y. Shiloach, New Connectivity and MSF algorithms for Shuffle-exchange and PRAM, *IEEE Trans. Computing*, **36** (1987) 1258-1263.

[2] P. Beame and J. Hastad, Optimal bounds for decision problems on the CRCW PRAM, *J. Assoc. Comput. Machinery*, **36** (1989) 643-670.

[3] C. Beeri, R. Fagin, D. Maier, M. Yannakakis, On the desirability of acyclic database schemes, *J. Assoc. Comput. Machinery*, **30** (1983) 479-513.

[4] N. Chandrasekharan and S.S. Iyengar, NC algorithms for recognizing chordal graphs and k-trees, *IEEE Trans. Computers*, **37** (1988) 1178-1183.

[5] N. Chandrasekharan, V.S. Lakshmanan and M. Medidi, Efficient parallel algorithms for finding chordless cycles in graphs, *Parallel. Proc. Letters*, **3** (1993) 165-170.

[6] A. Coppersmith and S. Winograd, Matrix multiplication via arithmetic progression, Proc. 19th Ann. ACM Symp. on Theory of Computing (STOC'87) 1-6.

[7] A. Edenbrandt, Chordal graph recognition is in NC, *Inf. Proc. Letters*, **24** (1987) 239-241.

[8] F.E. Fich, P.L. Ragde and A. Wigderson, Relations between concurrent-write models of parallel computation, in *Proc. 3rd Annual ACM Symposium on Principles of Distributed Computing*, 1984, 179-189.

[9] D.R. Fulkerson, O.A. Gross, Incidence matrices and interval graphs, *Pacific J. Mathematics*, **15** (1965) 835-855.

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

[11] R.B. Hayward, Weakly triangulated graphs, *J. Comb. Theory, B*, **39** (1985) 200-208.

[12] C-W. Ho and R.C.T. Lee, Efficient parallel algorithms for maximal cliques, clique tree, and minimum colouring on chordal graphs, *Inf. Proc. Letters*, **28** (1988) 301-309.

[13] C-W. Ho and R.C.T. Lee, Counting clique trees and computing perfect elimination schemes in parallel, *Inform. Process. Letters*, **31** (1989) 61-68.

[14] P.N. Klein, Efficient parallel algorithms for chordal graphs, in *Proc. 29th IEEE Symp. on Foundations of Computer Science*, 1988, 150-161.

[15] J. Naor, M. Naor and A. Schaffer, Fast parallel algorithms for chordal graphs, *SIAM J. Computing*, **18** (1989) 327-349.

[16] S.D. Nikolopoulos and S.D. Danielopoulos, Parallel computation of perfect elimination schemes using partition techniques on triangulated graphs, *Computers and Math. with Applications*, **29** (1995) 47-57.

[17] D.J. Rose, E. Tarjan and G.S. Luerer, Algorithmic aspects of vertex elimination on graphs, *SIAM J. Computing*, **5** (1976) 266-283.

[18] D.J. Rose, Triangulated graphs and the elimination process, *J. Math. Anal. Applications*, **32** (1970) 597-609.

[19] Y. Shiloach and U. Vishkin, An $O(log n)$ parallel connectivity algorithm, *J. of Algorithms*, **3** (1982) 57-67.

[20] J.P. Spinrad, Finding large holes, *Inf. Proc. Letters,* **39** (1991) 227-229.

[21] R. Sritharan and J.P. Spinrad, Algorithms for weakly triangulated graphs, *Manuscript,* 1992.

[22] R.E. Tarjan and M. Yannakakis, Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, *SIAM J. Computing,* **13** (1984) 566-579.