A faster parallel connectivity algorithm on cographs

Sun-Yuan Hsieh

Department of Computer Science and Information Engineering, National Cheng Kung University, No. 1, University Road, Tainan 70101, Taiwan

Received 6 March 2006; received in revised form 12 May 2006; accepted 30 May 2006

Abstract

Cographs are a well-known class of graphs arising in a wide spectrum of practical applications. In this note, we show that the connected components of a cograph $G$ can be optimally found in $O(\log \log \log \Delta(G))$ time using $O\left( \frac{(n+m)}{\log \log \log \Delta(G)} \right)$ processors on a common CRCW PRAM, or in $O(\log \Delta(G))$ time using $O\left( \frac{(n+m)}{\log \Delta(G)} \right)$ processors on an EREW PRAM, where $\Delta(G)$ is the maximum degree of $G$, and $n$ and $m$ respectively are the numbers of vertices and edges of $G$. These are faster than the previously best known result on general graphs.

Keywords: Parallel algorithms; Graph algorithms; Cographs; Parallel random access machine; PRAM

1. Introduction

A well-known class of graphs arising in a wide spectrum of practical applications is the class of cographs or complement reducible graphs. Cographs were introduced in the early 1970s by Lerchs [17]. Names synonymous with cographs include $D^*$-graphs, $P_4$-restricted graphs, and hereditary Dacey graphs defined in the study of empirical logic [23]. Cographs form a subclass of perfect graphs [8] that are graphs $G$ in which the maximum clique size equals the chromatic number for every induced subgraph of $G$ [5,11]. Furthermore, cographs are precisely the graphs which contain no induced subgraph isomorphic to $P_4$ (the chordless path on four vertices). The cographs have been studied extensively from both the theoretical and algorithmic points of view [1–4,8–10,12,16–22].

Given a graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, let $N_G(v) = \{u \in V : (u, v) \in E\}$ and $\Delta(G) = \max_{v \in V(G)} |N_G(v)|$. In this note, we show that the connected components of a cograph can be optimally found in $O(\log \log \log \Delta(G))$ time using $O\left( \frac{(n+m)}{\log \log \log \Delta(G)} \right)$ processors on a common CRCW PRAM, or in $O(\log \Delta(G))$ time using $O\left( \frac{(n+m)}{\log \Delta(G)} \right)$ processors on an EREW PRAM. Both are faster than the previously best known results on general graphs, that take $O(\log^2 n)$ time using $O\left( \frac{n^2}{\log n} \right)$ processors on a common CRCW PRAM for dense graphs [14], $O(\log n)$ time using $O(n+m)$ processors on an arbitrary CRCW PRAM for sparse graphs [14], $O(\log^2 n)$ time using $O(n+m)$ processors on a CREW PRAM [13], and $O(\log n \log \log n)$ time using $O(n+m)$ processors on...
an EREW PRAM [7]. Finding connected components of a cograph has applications for cograph recognition and
distance-hereditary graph recognition [10].

The computation model used here is the deterministic parallel random access machine (PRAM) which permits
concurrent read and write (CRCW) in its shared memory, or exclusive read and write (EREW) in its shared
memory [15] (see also [14]). In particular, the common CRCW PRAM allows concurrent writes only when all
processors are attempting to write the same value. The arbitrary CRCW PRAM allows an arbitrary processor to
succeed [15] when all processors are attempting to write the same value.

2. Preliminaries

This work considers finite, simple and loopless graphs \( G = (V, E) \), where \( V \) and \( E \) are the vertex and edge sets
of \( G \), respectively. Let \( n = |V| \) and \( m = |E| \). For convenience, we also use \( V(G) \) and \( E(G) \) to denote the vertex
and edge sets of \( G \), respectively. For an undirected graph, the edge joining \( x \) and \( y \) is denoted by \( xy \). For two graphs
\( G_1 = (V_1, E_1) \) and \( G_2 = (V_2, E_2) \), the union of \( G_1 \) and \( G_2 \), denoted by \( G_1 \cup G_2 \), is the graph \((V_1 \cup V_2, E_1 \cup E_2)\). A
subgraph of \( G = (V, E) \) is a graph \((V', E')\) such that \( V' \subseteq V \) and \( E' \subseteq E \). An induced subgraph is an edge-
preserving subgraph, that is, \((V', E')\) is an induced subgraph of \((V, E)\) iff \( V' \subseteq V \) and \( E' = \{(x, y) \in E : x, y \in V'\} \). Let \( G[X] \) denote the subgraph of \( G \) induced by \( X \subseteq V \). For graph-theoretic terminologies and notation not mentioned
here, readers should refer to [24].

A path in a graph \( G \) is a sequence of distinct vertices \( x_1, x_2, \ldots, x_k \) such that \( x_i x_{i+1} \in E(G) \) for \( i = 1, 2, \ldots, k-1 \). A path \( x_1, x_2, \ldots, x_k \) is chordless if \( x_i x_j \not\in E(G) \) for any two non-consecutive vertices \( x_i, x_j \) in the path. Throughout
this chordless path on four vertices is denoted by \( P_4 \). A graph is said to be \( P_4 \)-free if the graph contains no
induced subgraph isomorphic to a \( P_4 \). The distance of vertices \( x \) and \( y \) of \( G \), denoted by \( \dist(x, y) \), is the length of
a shortest path between \( x \) and \( y \) in \( G \). A graph \( G \) is connected iff for each pair of vertices \( x \) and \( y \), there is a path
between \( x \) and \( y \) in \( G \). A connected component of \( G \) is the vertex set of a maximal connected subgraph.

**Definition 1 ([17]).** The class of cographs is defined recursively as follows:
1. a single-vertex graph is a cograph;
2. the disjoint union of cographs is a cograph;
3. the complement of a cograph is a cograph.

3. A triply logarithmic time for finding connected components

In this section, we present a method for finding connected components of a cograph in triply logarithmic time using
totally a linear number of operations.

We begin with a few definitions. Let \( G = (V, E) \) be an undirected graph. Two vertices \( u \) and \( v \) are connected if
\( u = v \) or there exists a path \( u = u_1, u_2, \ldots, u_k = v \). This relation is clearly an equivalence relation on \( V \), and hence partitions
\( V \) into equivalence classes \( V_1, V_2, \ldots, V_j \). Clearly, each \( V_j \), \( 1 \leq j \leq j \), is a connected component of \( G \).

A rooted–directed tree \( T \) is a directed graph with special vertex \( r \) such that (1) every \( v \in V - \{r\} \) has outdegree 1,
and the outdegree of \( r \) is 0, and (2) for every \( v \in V - \{r\} \), there exists a directed path from \( v \) to \( r \). The special vertex \( r \)
called the root of \( T \). In particular, if each vertex is directly connected to the root \( r \), then the corresponding directed
tree is called a rooted star.

A pseudoforest is a directed graph in which each vertex has an outdegree less than or equal to 1. For a vertex
\( x \in V(G) \), let \( \dist_2(x) = \{y \in V(G) | 0 \leq \dist(x, y) \leq 2\} \). For a given \( n \)-vertex graph \( G \), assume that
the vertices of \( G \) are represented by \( 1, 2, \ldots, n \) in this section. Define the selection function \( f : V \to V \) by
\( f(v) = \min\{|u| u \in \dist_2(v)\} \). Let \( (u, v) \) be a directed edge pointed from \( u \) to \( v \). Note that the above function defines a pseudoforest \((V, F)\), where \( F = \{(v, f(v)) | v \in V\} \). For convenience, we shall refer to these structures as directed
trees or rooted stars although the degree of each root is nonzero.

**Lemma 1.** Let \( G = (V, E) \) be a cograph, and let \( f : V \to V \) be the selection function that defines a pseudoforest
\( \mathcal{F} \) for partitions of \( V \) into \( V_1, V_2, \ldots, V_i \), where each \( V_i \) is the set of vertices in a directed tree \( T_i \) of \( \mathcal{F} \). Then, the
following conditions hold:
1. All the vertices in each \( V_i \) form a connected component of \( G \).
2. Each $T_i$ is a rooted star.
3. Each cycle in $F$ is a self-loop.
4. The root of each tree $T_i$ in $F$ is the smallest vertex of $T_i$.

Proof. To show that Claim 1 holds, we first show that all the vertices in each $V_i$ belong to a connected component $C$ of $G$. Without loss of generality, assume that $|V_i| \geq 2$, and let $u$ and $v$ be two distinct vertices of $V_i$. Let $T'_i$ be the underlying undirected graph of $T_i$ by regarding each arc as an undirected edge. Since there is a unique path between $u$ and $v$ in $T'_i$, a path between $u$ and $v$ exists in $G$. Hence all the vertices in $V_i$ belong to $C$ of $G$. We next show that $V_i = C$, i.e., there is no other $V_j$, $j \neq i$, belonging to $C$. Assume, by contradiction, that there are $V_{j_1}, V_{j_2}, \ldots, V_{j_t}$, belonging to $C$, where $s \geq 2$. Let $w = \min\{x | x \in \bigcup_{i=1}^{t} V_{j_i}\}$. Since $G$ is $P_4$-free and $C$ is a connected component of $G$, all the vertices $v \in (\bigcup_{i=1}^{t} V_{j_i} \setminus \{w\})$ have $1 \leq \text{dist}(v, w) \leq 2$. By the definition of the selection function, all the above $s$ trees must join together to form one tree. This contradicts the assumption.

Clearly, Claim 2 holds by the definition of the selection function and the fact that cographs are $P_4$-free.

We now establish Claims 3 and 4. By Claim 2, each $T_i$ is a rooted star. Thus the root $r$ is the smallest vertex in $T_i$. By the definition of the selection function, $f(r) = r$, that forms a self-loop. □

The following lemma is useful for implementing the selection function.

Lemma 2. Given $n$ elements from the domain $\{1, 2, \ldots, n\}$, the minimum element can be found with the following parallel complexities:

1. [6] $O(\log \log n)$ time using $O\left(\frac{n}{\log \log \log n}\right)$ processors on a common CRCW PRAM.
2. [14] $O(\log n)$ time using $O\left(\frac{n}{\log n}\right)$ processors on an EREW PRAM.

Lemma 3. Let $G = (V, E)$ be a cograph. The selection function $f : V \to V$ can be implemented in $O(\log \log \Delta(G))$ time using $O\left(\frac{(n+m)}{\log \log \log \Delta(G)}\right)$ processors on a common CRCW PRAM, or in $O(\log \Delta(G))$ time using $O\left(\frac{(n+m)}{\log \Delta(G)}\right)$ processors on an EREW PRAM.

Proof. Assume that the vertices of $G$ are represented by $n$ positive integers $1, 2, \ldots, n$. Algorithm 1 shows how to construct the selection function.

```
Algorithm 1 Selection(min_adj_value(v); f(v))
1: for each vertex v \in V do
2:   min_adj_value(v) \leftarrow v    /* Initialization */
3:   end for
4:   for i = 1 to 2 do
5:     for each vertex v \in V do
6:       min_adj_value(v) \leftarrow \min\{min_adj_value(u) | u \in N_G[v]\}
7:     end for
8:   end for
9:   for each vertex v \in V do
10:  f(v) \leftarrow \min_adj_value(v)
11: end for
```

Since the distance between the minimum vertex in a connected component $C$ and each of the others in $C$ is at most two, it is not difficult to verify that the above algorithm correctly constructs a pseudoforest defined by the selection function $f : V \to V$.

We next show the time–processor complexity of the algorithm. We assume that the input graph $G$ is given in adjacency list representation. Clearly, lines 1–3 can be implemented in $O(1)$ time using $O(n)$ processors on an EREW PRAM. As with the aid of Brent’s scheduling principle [15], this can be scheduled to achieve the required parallel complexities. By Lemma 2, lines 4–8 can be implemented in $O(\log \log \Delta(G))$ time using in total $O\left(\frac{m}{\log \log \log \Delta(G)}\right)$ processors on a common CRCW PRAM, or in $O(\log \Delta(G))$ time using in total $O\left(\frac{m}{\log \Delta(G)}\right)$ processors on an EREW PRAM. Clearly, lines 9–11 can be implemented in $O(1)$ time using $O(n)$ processors on an EREW PRAM. Therefore, the algorithm runs on a CRCW or EREW PRAM with the desired complexities. □
By Lemmas 1–3, we have the following theorem.

**Theorem 1.** The connected components of a cograph can be found in \(O(\log \log \log \Delta(G))\) time using \(O\left(\frac{n+m}{\log \log \log \Delta(G)}\right)\) processors on a common CRCW PRAM, or in \(O(\log \Delta(G))\) time using \(O\left(\frac{n+m}{\log \Delta(G)}\right)\) processors on an EREW PRAM.

**References**