LBFS orderings and cocomparability graphs

Derek G. Corneil*  Stephan Olariu†  Lorna Stewart‡

Abstract
Recently Lexicographic Breadth First Search (LBFS) has received considerable attention and has been shown to be very useful in the design of simple linear time algorithms for a number of diverse applications. In light of the popularity of LBFS, it is somewhat surprising that little work has been done on the study of LBFS orderings themselves. In this paper, we provide characterizations of LBFS orderings and show the impact of these results.

1 Introduction.
Lexicographic Breadth First Search (LBFS) was first introduced in the classical paper by Rose, Tarjan and Lueker [5] to provide a simple linear time algorithm to recognize chordal graphs (graphs with no induced cycle of size greater than 3). We reproduce below the details of LBFS from [5] (slightly modified).

Algorithm 1.1. LBFS(G, x)
{Input: a connected graph G = (V, E) and a distinguished vertex x of G;
Output: a numbering σ of the vertices of G}
begin
label(x) ← |V|
for each vertex v in V − {x} do
label(v) ← λ
for i ← |V| downto 1 do begin
v ← an unnumbered vertex with (lexicographically) the largest label
σ(v) ← |V| + 1 − i;
{assign to v number |V| + 1 − i}
for each unnumbered vertex u in N(v) do
append i to label(u)
end {LBFS}

Algorithm 1.1 numbers vertices from 1 to |V| whereas the original LBFS algorithm of [5] numbers vertices from |V| down to 1. Consequently, the output produced by Algorithm 1.1 is exactly the reverse ordering of the output of the original LBFS algorithm. We shall say that u occurs before v in σ, or u occurs to the left of v in σ, or u <σ v, whenever σ(u) < σ(v). We use the terms after, to the right of, and >σ, analogously.

Since the appearance of the Rose, Tarjan and Lueker paper, a great deal of research has been done on other applications of LBFS including the recognition of various graph families, determining dominating pairs and vertices of high eccentricity. One of the most surprising such results is the simple linear time 4-sweep LBFS recognition algorithm for interval graphs [1]. In light of this result it is natural to ask whether a similar LBFS-based algorithm exists for other graph families, most notably for cocomparability graphs (whose complements admit transitive orientations). In order to get more insight into this question we first recall the following characterizations of interval graphs and cocomparability graphs. In the remainder of this note, an ordering satisfying Theorem 1.1 will be called an O-ordering, and the ordering implied by Proposition 1.1 will be termed a KS-ordering.

Theorem 1.1. [4] A graph G = (V, E) is an interval graph if and only if there exists a linear order < on the set of its vertices such that for every choice of vertices u, v, w, with u < v and v < w, uw ∈ E implies vw ∈ E.

Proposition 1.1. [3] A graph G = (V, E) is a cocomparability graph if and only if there exists a linear order < on the set of its vertices such that for every choice of vertices u, v, w, with u < v and v < w, uw ∈ E implies uw ∈ E or vw ∈ E.

2 Characteristics of LBFS orderings.
We start by stating a characterization of when a given ordering is an LBFS ordering.

Theorem 2.1. [2] Let G = (V, E) be an arbitrary graph and σ an ordering of V. σ is an LBFS order if and only if for all a, b, c satisfying a <σ b, b <σ c, ac ∈ E, and ab ∉ E: there exists d satisfying d <σ a, db ∈ E, and dc ∉ E.
In a vertex ordering \( \sigma \) which is not an LBFS ordering, a triple of vertices \( a, b, c \) that violates the condition of Theorem 2.1 is referred to as a bad triple; any other triple is termed a good triple.

It is interesting to see what this rather simple theorem says about interval graphs.

**Corollary 2.1.** Every O-ordering of an interval graph is a LBFS ordering.

**Proof.** Let \( \sigma \) be an O-ordering that is not an LBFS ordering and let \( (a, b, c) \) be a bad triple. Since \( ab \notin E \) we immediately see that \( \sigma \) is not an O-ordering. □

We now examine an arbitrary ordering that is not an LBFS ordering and show that a carefully chosen bad triple has a very special property.

**Theorem 2.2.** Let \( G = (V, E) \) be an arbitrary graph and \( \sigma \) an ordering of \( V \) which is not an LBFS ordering.

Let \( a \) be the leftmost vertex in a bad triple;

Let \( c \) be the leftmost vertex that is the third vertex in a bad triple having \( a \) as the first vertex;

Let \( b \) be the rightmost vertex that is the second vertex in a bad triple having \( a \) as the first vertex and \( c \) as the third vertex.

Then \( b \) and \( c \) are consecutive in \( \sigma \).

**Proof.** Let \( c' \) be an arbitrary vertex between \( b \) and \( c \). First we show that \( ac' \notin E \). Suppose to the contrary. By the choice of \( c \), \( (a, b, c') \) is a good triple and thus there exists vertex \( d \), chosen to be as leftmost as possible, such that \( db \in E \) and \( dc' \notin E \). Since \( (a, b, c) \) is a bad triple, \( dc \in E \). Now, by the choice of \( a \), \( (d, c', e) \) is a good triple and thus vertex \( e \) exists with \( e <_\sigma d \) where \( ec' \notin E \) and \( ec \notin E \). Since \( (a, b, c) \) is a bad triple, \( eb \notin E \). By the choice of \( a \), \( (e, b, c') \) is not a bad triple and thus there is a vertex \( f \) such that \( f <_\sigma e \), \( fb \in E \) and \( fe' \notin E \). But \( f \) contradicts the fact that \( d \) was chosen to be leftmost.

We now show that \( c' \) cannot exist. By the choice of \( b \), \( (a, c', d) \) is a good triple and thus there is a vertex \( d \), chosen to be as leftmost as possible, such that \( dd' \in E \), \( dc' \notin E \), which in turn implies that \( db \notin E \) since \( (a, b, c) \) is a bad triple. By the choice of \( a \), \( (d, b, c') \) is a good triple and thus there exists \( e <_\sigma d \) where \( eb \in E \) and \( ec' \notin E \) again implying that \( ec \notin E \) (since \( (a, b, c) \) is bad). Finally, by the choice of \( a \), \( (c, c', e) \) is a good triple and thus there exists \( f <_\sigma e \) such that \( fe' \notin E \), and \( fe \notin E \) but this contradicts the choice of \( d \) being leftmost. □

**Corollary 2.2.** If \( G \) is a cocomparability graph, then it has a KS-ordering that is a LBFS.

**Proof.** Assume the statement is false and let \( G \) be a counterexample with \( \sigma \) a KS-ordering that has as few bad triples (LBFS) as possible. Choose the bad triple \( (a, b, c) \) as in the statement of Theorem 2.2. Immediately we see that \( b \) and \( c \) are consecutive in \( \sigma \) (by Theorem 2.2) and that \( bc \in E \) (since \( \sigma \) is a KS-ordering), \( ab \notin E \) and \( ac \notin E \).

Now consider the ordering \( \tau \) which is identical to \( \sigma \) except that \( b \) and \( c \) have been interchanged. We claim that \( \tau \) is a KS-ordering with fewer bad triples than \( \sigma \) thus contradicting our choice of \( \sigma \).

To see that \( \tau \) is a KS-ordering, we suppose not and let triple \( (x, y, z) \) occur in \( \tau \) where \( x <_\tau y <_\tau z \), \( xz \in E \), \( xy \notin E \) and \( yz \notin E \). Since \( \sigma \) is a KS-ordering, this clearly implies that either \( b = z \) or \( c = x \). In both cases, the fact that \( \sigma \) is a KS-ordering implies that \( y = c \) (if \( b = z \)) or \( y = b \) (if \( c = z \)). But now the edge \( bc \) contradicts the subgraph induced on \( (x, y, z) \).

Since the bad triple \( (a, b, c) \) in \( \sigma \) does not occur in \( \tau \), it suffices to prove that every bad triple in \( \tau \) is also a bad triple in \( \sigma \). Suppose not and let \( (x, y, z) \) be a bad triple in \( \tau \) that is not in \( \sigma \).

We proceed by looking at possible locations for \( x \) in \( \tau \). If \( x <_\tau a \), then by the choice of \( (a, b, c) \) in \( \sigma \), \( z = b \) and \( y = c \); but now \( (a, b, c) \) is not a bad triple in \( \sigma \). If \( x = a \), then an easy argument shows that \( (x, y, z) \) is a bad triple in \( \sigma \). Finally \( a <_\tau x \); clearly \( x <_\tau c \) since otherwise \( (x, y, z) \) is bad in \( \sigma \). Similarly we see that \( z = b \) and \( y = c \). But now \( (x, y, z) \) is not bad in \( \sigma \) since \( ay \in E \) and \( az \notin E \). □

**Corollary 2.2** motivates a search for a multi-sweep LBFS algorithm to recognize cocomparability graphs. This promises to be a tantalizing topic for further work.

**References**


