

Graph Searches and Cocomparability Graphs

Derek Corneil¹ Jeremie Dusart, Michel Habib, Ekki Koehler

¹Computer Science, University of Toronto

CanaDAM
June 10, 2013

- Maximal (Maximum?) Independent Set Algorithm
- Cocomparability Graphs, Comparability Graphs and Posets
- Overview of Graph Searching
- Comments, New Results and Concluding Remarks

Maximal (Maximum?) Independent Set Algorithm

Maximal (Maximum?) Independent Set (MIS):

Input: A connected graph $G = (V, E)$ and vertex ordering σ

Output: Set I containing the vertices of an IS

$I \leftarrow \emptyset$; $V' \leftarrow V$ { V' stores the unprocessed vertices}; $j \leftarrow 0$;

while $V' \neq \emptyset$ **do**

$j \leftarrow j + 1$;

$x_j \leftarrow$ the rightmost vertex of V' , as ordered by σ ;

$I \leftarrow I \cup \{x_j\}$; $V' \leftarrow V' \setminus N[x_j]$;

end

return (I)

Questions about the algorithm

- How is σ determined?

Questions about the algorithm

- How is σ determined?
- *To be discussed later.*

Questions about the algorithm

- How is σ determined?
- *To be discussed later.*
- In general can I be “close to” a maximum independent set?

Questions about the algorithm

- How is σ determined?
- *To be discussed later.*
- In general can I be “close to” a maximum independent set?
- *NO. Consider $K_{2,n}$ and a degree n vertex being the rightmost vertex of σ . $I =$ the two degree n vertices; whereas the maximum IS consists of the n degree 2 vertices.*

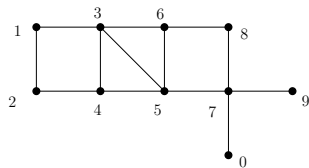
Questions about the algorithm

- How is σ determined?
- *To be discussed later.*
- In general can I be “close to” a maximum independent set?
- *NO. Consider $K_{2,n}$ and a degree n vertex being the rightmost vertex of σ . $I =$ the two degree n vertices; whereas the maximum IS consists of the n degree 2 vertices.*
- How could we “Certify” that I is a maximum IS?

Questions about the algorithm

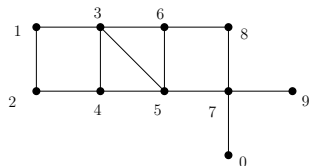
- How is σ determined?
- *To be discussed later.*
- In general can I be “close to” a maximum independent set?
- *NO. Consider $K_{2,n}$ and a degree n vertex being the rightmost vertex of σ . $I =$ the two degree n vertices; whereas the maximum IS consists of the n degree 2 vertices.*
- How could we “Certify” that I is a maximum IS?
- *Find a clique cover of the same cardinality. Note that for any graph G , $\alpha(G) \leq \kappa(G)$ where α is the maximum cardinality IS and κ is the minimum cardinality clique cover. If the graph is perfect, then equality holds.*

Example



- $\sigma = 0798653421$

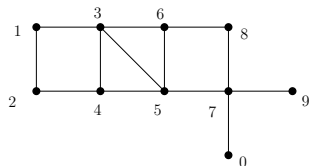
Example



- $\sigma = 0798653421$

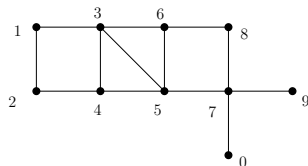
- $\sigma = 0798654$ **1**

Example



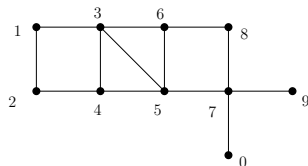
- $\sigma = 0798653421$
- $\sigma = 0798654$ **1**
- $\sigma = 07986$ **4** **1**

Example



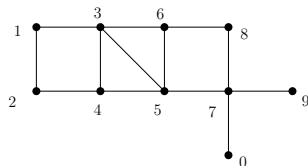
- $\sigma = 0798653421$
- $\sigma = 0798654\mathbf{1}$
- $\sigma = 07986\mathbf{41}$
- $\sigma = 079\mathbf{641}$

Example

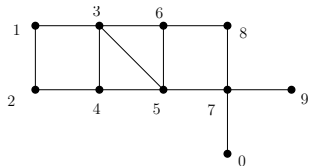


- $\sigma = 0798653421$
- $\sigma = 0798654\mathbf{1}$
- $\sigma = 07986\mathbf{41}$
- $\sigma = 079\mathbf{641}$
- $\sigma = 0\mathbf{9641}$

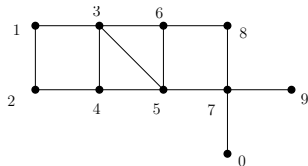
Example



- $\sigma = 0798653421$
- $\sigma = 0798654\mathbf{1}$
- $\sigma = 07986\mathbf{41}$
- $\sigma = 079\mathbf{641}$
- $\sigma = 0\mathbf{9641}$
- $\sigma = I = 0\mathbf{9641}$

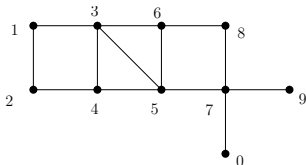


• $\sigma = l = 0\ 9\ 6\ 4\ 1$



- $\sigma = l = 0\ 9\ 6\ 4\ 1$

- $\sigma = 0\ 7\ 9\ 8\ 6\ 5\ 3\ 4\ 2\ 1$



- $\sigma = I = 0\ 9\ 6\ 4\ 1$
- $\sigma = 0\ 7\ 9\ 8\ 6\ 5\ 3\ 4\ 2\ 1$
- Note that I is of maximum cardinality, as shown by the clique cover (from right to left): $\{2, 1\}, \{5, 3, 4\}, \{8, 6\}, \{7, 9\}, \{0\}$

Cocomparability Graphs, Comparability Graphs and Posets

- A **cocomparability graph** $G(V, E)$ is one where the complement graph (known as a **comparability graph**) has a transitive orientation of its edges.

Cocomparability Graphs, Comparability Graphs and Posets

- A **cocomparability graph** $G(V, E)$ is one where the complement graph (known as a **comparability graph**) has a transitive orientation of its edges.
- In particular, there is an orientation of \overline{E} such that if there is an arc from x to y and an arc from y to z , then there is an arc from x to z .

Cocomparability Graphs, Comparability Graphs and Posets

- A **cocomparability graph** $G(V, E)$ is one where the complement graph (known as a **comparability graph**) has a transitive orientation of its edges.
- In particular, there is an orientation of \overline{E} such that if there is an arc from x to y and an arc from y to z , then there is an arc from x to z .
- A comparability graph together with an acyclic transitive orientation of its edges can be equivalently represented by a partially ordered set (also called a **poset**).

Cocomparability Graphs, Comparability Graphs and Posets

- A **cocomparability graph** $G(V, E)$ is one where the complement graph (known as a **comparability graph**) has a transitive orientation of its edges.
- In particular, there is an orientation of \bar{E} such that if there is an arc from x to y and an arc from y to z , then there is an arc from x to z .
- A comparability graph together with an acyclic transitive orientation of its edges can be equivalently represented by a partially ordered set (also called a **poset**).
- A poset consists of a set V together with an irreflexive, antisymmetric and transitive binary relation $<$ that imposes a “precedes” relationship on certain pairs of elements of V . Two elements $x, y \in V$ are said to be **comparable** if $x < y$, or $y < x$; otherwise the elements are called **incomparable**. A **linear extension** of a poset is a total ordering of V that respects the ordering of all comparable pairs.

Definitions Continued

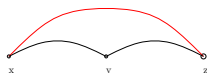
- DEFINITION: $G(V, E)$ is a **interval graph** if it is the intersection graph of subpaths of a path; namely, each vertex represents a subpath and two vertices are adjacent iff their subpaths intersect. Note that interval graphs are a strict subset of cocomparability graphs.

Definitions Continued

- DEFINITION: $G(V, E)$ is a **interval graph** if it is the intersection graph of subpaths of a path; namely, each vertex represents a subpath and two vertices are adjacent iff their subpaths intersect. Note that interval graphs are a strict subset of cocomparability graphs.
- THEOREM: G is interval iff there is an ordering of V such that for all $x < y < z, xz \in E$ implies $xy \in E$ (**I ORDER**)

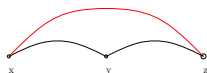
Definitions Continued

- DEFINITION: $G(V, E)$ is a **interval graph** if it is the intersection graph of subpaths of a path; namely, each vertex represents a subpath and two vertices are adjacent iff their subpaths intersect. Note that interval graphs are a strict subset of cocomparability graphs.
- THEOREM: G is interval iff there is an ordering of V such that for all $x < y < z$, $xz \in E$ implies $xy \in E$ (**I ORDER**)
- THEOREM: G is cocomparability iff there is an ordering of V such that for all $x < y < z$, $xz \in E$ implies $xy \in E$ OR $yz \in E$ OR both (**COCOMP order**)



Definitions Continued

- DEFINITION: $G(V, E)$ is a **interval graph** if it is the intersection graph of subpaths of a path; namely, each vertex represents a subpath and two vertices are adjacent iff their subpaths intersect. Note that interval graphs are a strict subset of cocomparability graphs.
- THEOREM: G is interval iff there is an ordering of V such that for all $x < y < z$, $xz \in E$ implies $xy \in E$ (**I ORDER**)
- THEOREM: G is cocomparability iff there is an ordering of V such that for all $x < y < z$, $xz \in E$ implies $xy \in E$ OR $yz \in E$ OR both (**COCOMP order**)



- Note that every I ORDER serves as an appropriate σ for the MIS algorithm; but many COCOMP orders fail.

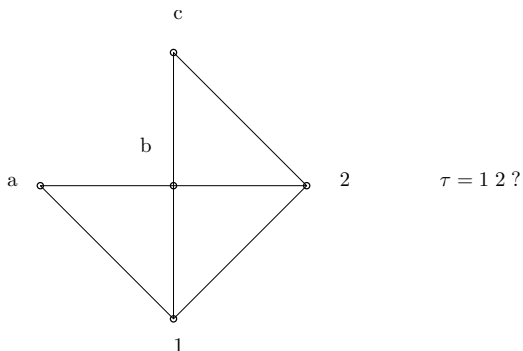
Overview of Graph Searches

- algorithms for visiting all vertices and edges of a given graph
- BFS and DFS discovered in the late 1890s for maze traversal.
- In the 1960s and 1970s BFS and DFS were shown to have many applications in computer science.
- In 1976 Rose, Tarjan and Lueker presented LBFS as a way of recognizing chordal graphs (no induced cycle of size greater than 3).
- Many applications of LBFS were later found including a linear time algorithm for recognizing interval graphs.
- There is a vertex ordering characterization of LBFS orderings [Golumbic; Brandstadt, Dragan and Nicolai].
- The study of such VOCs lead to the discovery of LDFS [C. and Krueger].

Lexicographic Depth First Search (LDFS)

Roughly speaking, LDFS is a DFS where ties are broken by favouring vertices with adjacencies to latest visited vertices.

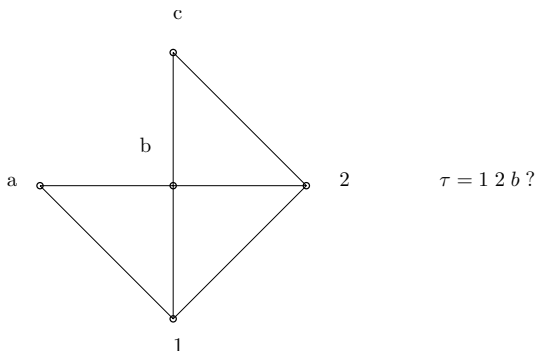
COMPLETE the LDFS



Lexicographic Depth First Search (LDFS)

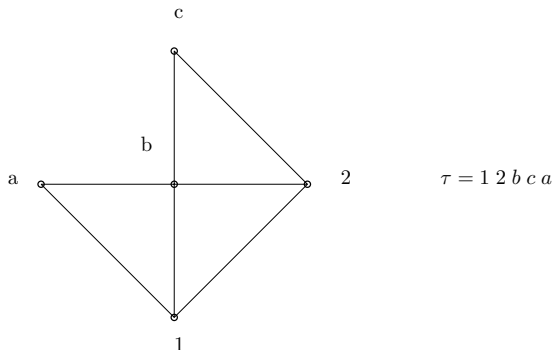
Roughly speaking, LDFS is a DFS where ties are broken by favouring vertices with adjacencies to latest visited vertices.

COMPLETE the LDFS



Lexicographic Depth First Search (LDFS)

COMPLETE the LDFS



IMPLEMENTATION: $O(\min\{n^2, n + m \log \log n\})$ Spinrad and ???

- LBFS cocomp orders may fail for the MIS algorithm; for example an LBFS starting at a degree n vertex in $K_{2,n}$.

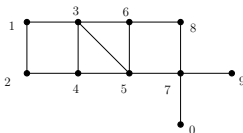
- LBFS cocomp orders may fail for the MIS algorithm; for example an LBFS starting at a degree n vertex in $K_{2,n}$.
- BUT LDFS cocomp orders succeed for the algorithm (including finding an optimum clique cover) - a surprisingly easy proof.

- LBFS cocomp orders may fail for the MIS algorithm; for example an LBFS starting at a degree n vertex in $K_{2,n}$.
- BUT LDFS cocomp orders succeed for the algorithm (including finding an optimum clique cover) - a surprisingly easy proof.
- How to generate an LDFS cocomp order? McConnell and Spinrad have a complicated linear time algorithm to generate τ , a cocomp order of a cocomp graph - but the current fastest algorithm to confirm that it is a cocomp order requires $O(MM)$ time.

- LBFS cocomp orders may fail for the MIS algorithm; for example an LBFS starting at a degree n vertex in $K_{2,n}$.
- BUT LDFS cocomp orders succeed for the algorithm (including finding an optimum clique cover) - a surprisingly easy proof.
- How to generate an LDFS cocomp order? McConnell and Spinrad have a complicated linear time algorithm to generate τ , a cocomp order of a cocomp graph - but the current fastest algorithm to confirm that it is a cocomp order requires $O(MM)$ time.
- Setting $\sigma = LDFS^+(G, \tau)$ yields an LDFS order that is a cocomp order if τ is a cocomp order.

- LBFS cocomp orders may fail for the MIS algorithm; for example an LBFS starting at a degree n vertex in $K_{2,n}$.
- BUT LDFS cocomp orders succeed for the algorithm (including finding an optimum clique cover) - a surprisingly easy proof.
- How to generate an LDFS cocomp order? McConnell and Spinrad have a complicated linear time algorithm to generate τ , a cocomp order of a cocomp graph - but the current fastest algorithm to confirm that it is a cocomp order requires $O(MM)$ time.
- Setting $\sigma = LDFS^+(G, \tau)$ yields an LDFS order that is a cocomp order if τ is a cocomp order.
- A $+$ sweep breaks ties by choosing the rightmost tied vertex as ordered by τ .

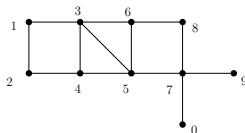
Example of LDFS⁺



Consider this graph and cocomp order $\tau = 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 0$

- $\sigma = 0\ 7\ 9\ 8\ 6\ 5\ 3\ 4\ 2\ 1$ - This ordering is the one used in the example of the MIS algorithm.

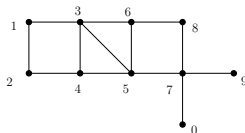
Example of LDFS⁺



Consider this graph and cocomp order $\tau = 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 0$

- $\sigma = 0\ 7\ 9\ 8\ 6\ 5\ 3\ 4\ 2\ 1$ - This ordering is the one used in the example of the MIS algorithm.
- The first vertex of σ is the rightmost vertex of τ , namely 0. The next vertex is 7.

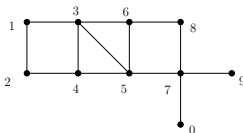
Example of LDFS⁺



Consider this graph and cocomp order $\tau = 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 0$

- $\sigma = 0\ 7\ 9\ 8\ 6\ 5\ 3\ 4\ 2\ 1$ - This ordering is the one used in the example of the MIS algorithm.
- The first vertex of σ is the rightmost vertex of τ , namely 0. The next vertex is 7.
- Now there is a tie amongst 8, 9, 5. Since 9 is rightmost it is chosen next, followed by 8 (rightmost between 5 and 8).

Example of LDFS⁺



Consider this graph and cocomp order $\tau = 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 0$

- $\sigma = 0\ 7\ 9\ 8\ 6\ 5\ 3\ 4\ 2\ 1$ - This ordering is the one used in the example of the MIS algorithm.
- The first vertex of σ is the rightmost vertex of τ , namely 0. The next vertex is 7.
- Now there is a tie amongst 8, 9, 5. Since 9 is rightmost it is chosen next, followed by 8 (rightmost between 5 and 8).
- The next vertex is 6 - now LDFS forces 5 and then 3, followed by 4, 2, 1.

- By having a certification step, we will either guarantee that we have a maximum IS or will output a message that the given ordering is not a cocomp ordering. Note that we do not confirm that our given ordering τ is a cocomp ordering.
- Similar LDFS⁺ modified interval graph algorithms work for:
 - Minimum Path Cover (equivalent to the bump number problem on posets) [C., Dalton, H.]
 - Longest Path [Mertzios, C.]
- These algorithms give us insight into the “LDFS structure of posets”.

- We have a characterization of the searches \mathcal{S} such that $\sigma = \mathcal{S}^+(\tau)$ is a cocomp order whenever τ is a cocomp order.

- We have a characterization of the searches \mathcal{S} such that $\sigma = \mathcal{S}^+(\tau)$ is a cocomp order whenever τ is a cocomp order.
- Using a new graph search we have an easier permutation recognition algorithm.

- We have a characterization of the searches \mathcal{S} such that $\sigma = \mathcal{S}^+(\tau)$ is a cocomp order whenever τ is a cocomp order.
- Using a new graph search we have an easier permutation recognition algorithm.
- We also have structural results on a lattice built on the set of maximal cliques in a cocomp graph - note that the number of such cliques can grow exponentially with n .

- We have a characterization of the searches \mathcal{S} such that $\sigma = \mathcal{S}^+(\tau)$ is a cocomp order whenever τ is a cocomp order.
- Using a new graph search we have an easier permutation recognition algorithm.
- We also have structural results on a lattice built on the set of maximal cliques in a cocomp graph - note that the number of such cliques can grow exponentially with n .
- Using these results we have a new graph search and simple algorithms to compute minimal clique separators and to find simplicial vertices in cocomp graphs.

Concluding Remarks

- Are there more new searches and new simple algorithms for cocomp graphs?

Concluding Remarks

- Are there more new searches and new simple algorithms for cocomp graphs?
- J.D., E.K and I are guardedly optimistic that we can extend Keil's HC algorithm for interval graphs to get a "linear" (off by a $\log \log n$ factor) HC algorithm for cocomp graphs. This algorithm is also certifying insofar as if there is no HC the algorithm outputs a toughness certificate.

Concluding Remarks

- Are there more new searches and new simple algorithms for cocomp graphs?
- J.D., E.K and I are guardedly optimistic that we can extend Keil's HC algorithm for interval graphs to get a "linear" (off by a $\log \log n$ factor) HC algorithm for cocomp graphs. This algorithm is also certifying insofar as if there is no HC the algorithm outputs a toughness certificate.
- New insights into the structure of posets? What about other areas of mathematics?

Concluding Remarks

- Are there more new searches and new simple algorithms for cocomp graphs?
- J.D., E.K and I are guardedly optimistic that we can extend Keil's HC algorithm for interval graphs to get a "linear" (off by a $\log \log n$ factor) HC algorithm for cocomp graphs. This algorithm is also certifying insofar as if there is no HC the algorithm outputs a toughness certificate.
- New insights into the structure of posets? What about other areas of mathematics?
- Can these results extend to AT-free graphs?

Concluding Remarks

- Are there more new searches and new simple algorithms for cocomp graphs?
- J.D., E.K and I are guardedly optimistic that we can extend Keil's HC algorithm for interval graphs to get a "linear" (off by a $\log \log n$ factor) HC algorithm for cocomp graphs. This algorithm is also certifying insofar as if there is no HC the algorithm outputs a toughness certificate.
- New insights into the structure of posets? What about other areas of mathematics?
- Can these results extend to AT-free graphs?
- Can we use graph searching for heuristic algorithms? Already used for the diameter of the giant component of the Facebook Graph.

Thank you for your attention, Eric