Simple vertex ordering characterizations for graph search (expanded abstract) [1]

Derek G. Corneil [2] and Richard Krueger [3]

Department of Computer Science
University of Toronto
Toronto, Ontario, Canada

Graph searching is fundamental. Many graph algorithms employ some mechanism for systematically visiting all vertices and edges of a given graph. After choosing some initial sequence of vertices, it is natural to choose as the next vertex to visit some vertex that is adjacent to a previously visited vertex whenever possible. This statement describes *graph search* in its most general setting. When selecting a next vertex to visit, there are typically many such vertices that may be chosen. Strategies such as *breadth-first search* (BFS) and *depth-first search* (DFS) can be employed to choose among these vertices.

Other strategies also exist. In the 1970s, Rose, Tarjan and Lueker [7] introduced a variant of BFS for determining perfect elimination orderings of chordal graphs. Their *lexicographic breadth-first search* (LexBFS) has garnered much attention recently and has become an important algorithm used for recognition of various graph families, diameter approximation, and finding key structures in certain graph classes [3]. One key point about LexBFS, and one of the reasons for its usefulness, is that there is a very simple characteri-

[1] This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Ontario Graduate Scholarship Program.

[2] Email: `dgc@cs.toronto.edu`

[3] Email: `krueger@cs.toronto.edu`

zation of LexBFS orderings [2]. This characterization is critical to the proofs of correctness of various algorithms. In light of the power of such a tool, it is surprising that the traditional searches of BFS and DFS are not known to have similar characterizations. Other than LexBFS, only the *maximum cardinality search* (MCS) of Tarjan and Yannakakis [8] has a similar known characterization [2].

We introduce new characterizations of vertex orderings for all known major search paradigms by examining a simple question. Suppose we are given a vertex ordering of graph G, and, in this ordering, vertex a is visited before vertex b, vertex b is visited before vertex c, and it happens that edge ac is present in G but edge ab is absent. We ask "how could a graph search visit vertex b before visiting vertex c?" With what is known so far, it seems that c should have been visited before b. What can we infer about the structure of G such that the search could have chosen b before c?

As mentioned, the answer to this question is known for LexBFS. There must exist another vertex d visited earlier than a such that db is an edge but dc is not an edge of G [5]. Brandstädt, Dragan and Nicolai [2] showed that this condition characterizes those vertex orderings that can be generated by LexBFS.

Experience with LexBFS has shown that such a characterizing property is invaluable when applying the search to various domains. Not only does this characterization simplify proofs of those graph properties that LexBFS can find, but it has led to algorithms using multiple LexBFS sweeps to reveal structures in various classes of graphs [3].

In this work we consider the other known search paradigms and derive characterizations of the vertex orderings obtainable under these paradigms. For notation, we suppose $\sigma = (v_1, v_2, \ldots, v_n)$ is an ordering of the vertices of a simple graph $G = (V, E)$ where $n = |V|$, and we say $v_i < v_j$ in σ if $i < j$. We say v_1 is the first vertex visited, and v_n the last; we caution the reader that some other papers may order vertices in the opposite direction.

We first consider generic graph search. The answer to our key question "how could b have been visited before c?" is simply that vertex b must have a neighbor d that is visited earlier than b. It turns out that this answer is also sufficient to characterize all graph search vertex orderings.

Theorem 1 *For an arbitrary graph G, an ordering σ of V is a search ordering of G if and only if σ has property (S).*

(S): Given an ordering σ of V, if $a < b < c$ and $ac \in E$ and $ab \notin E$, then there exists a vertex $d < b$ such that $db \in E$.

For BFS, the vertex ordering must respect the distance layering of vertices, in addition to being a search ordering and some other properties. Again the answer to our key question leads to a characterization of all orderings that can be found by BFS.

Theorem 2 *For an arbitrary graph G, an ordering σ of V is a BFS ordering of G if and only if σ has property (B).*

(B): Given an ordering σ of V, if $a < b < c$ and $ac \in E$ and $ab \notin E$, then there exists a vertex $d < a$ such that $db \in E$.

One notices that (B) is both a restriction of (S) and a generalization of the LexBFS property. Thus any LexBFS ordering is a BFS, and any BFS ordering is a graph search.

We now consider the other traditional search strategy, DFS. Opposite to BFS, at each step DFS will visit next a vertex that is adjacent to a vertex more recently visited. The answer to our key question captures this idea succinctly, and in a way fundamentally opposite to what we saw for BFS.

Theorem 3 *For an arbitrary graph G, an ordering σ of V is a DFS ordering of G if and only if σ has property (D).*

(D): Given an ordering σ of V, if $a < b < c$ and $ac \in E$ and $ab \notin E$, then there exists a vertex d with $a < d < b$ such that $db \in E$.

It has long been wondered whether there is a depth-first analogue to LexBFS. For the first time we now have an understanding of the difference between LexBFS and BFS, and DFS. Again examining our key question, we see that applying the same restriction to DFS as LexBFS applies to BFS results in a natural candidate for *lexicographic depth-first search* (LexDFS). The algorithm for LexDFS described in [4], [6] is very similar to the LexBFS algorithm, and yields the following characterization.

Theorem 4 *For an arbitrary graph G, an ordering σ of V is a LexDFS ordering of G if and only if σ has property (LD).*

(LD): Given an ordering σ of V, if $a < b < c$ and $ac \in E$ and $ab \notin E$, then there exists a vertex d with $a < d < b$ such that $db \in E$ and $dc \notin E$.

We now address the question of the relationship between LexDFS and LexBFS. What does the property that dc is not an edge give us? In fact, this very property was used by Tarjan and Yannakakis [8] to prove that LexBFS yields a perfect elimination ordering of a chordal graph, but they did not consider whether the property corresponded to a search algorithm. We introduce

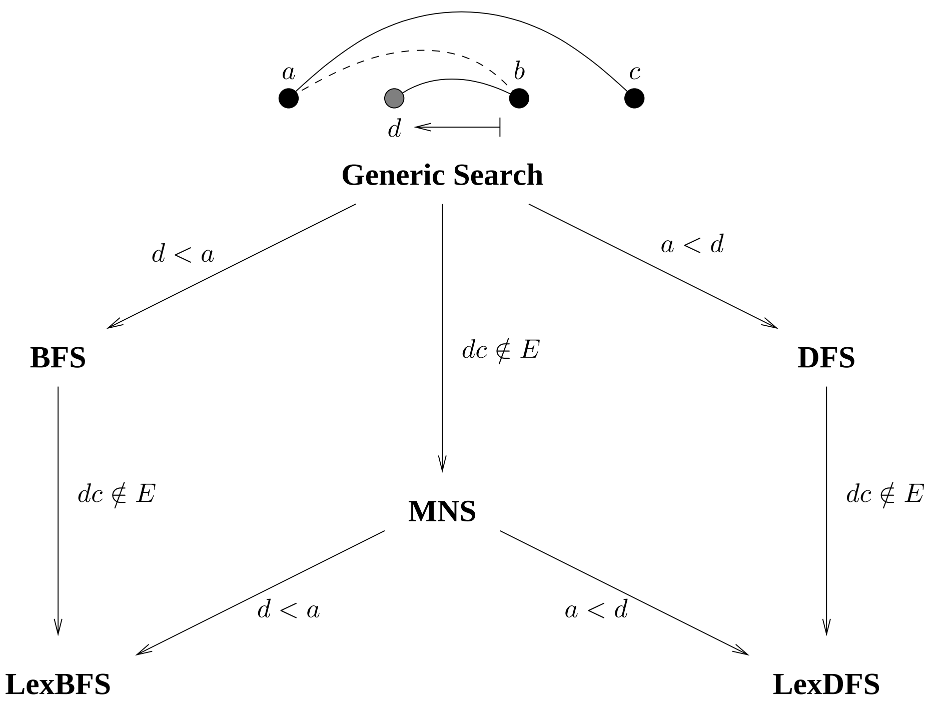

Fig. 1. Summary of search characterizations. For example, vertex ordering σ is a LexDFS ordering if and only if σ satisfies the condition for generic search together with $dc \notin E$ and $a < d$.

maximal neighborhood search (MNS) to complete our hierarchy of vertex ordering characterizations. The algorithm for MNS can be described as follows: at each step choose a vertex whose visited neighborhood (those neighbors which have already been visited) is maximal over all unvisited vertices.

Theorem 5 *For an arbitrary graph G, an ordering σ of V is a MNS ordering if and only if σ has property (M).*

(M): Given an ordering σ of V, if $a < b < c$ and $ac \in E$ and $ab \notin E$, then there exists a vertex d with $d < b$ and $db \in E$ and $dc \notin E$.

A vertex with lexicographically maximum label (as in LexBFS and LexDFS) or maximum cardinality neighborhood (as in MCS) will have a maximal neighborhood, thus these searches are restrictions of MNS (as also seen by the characterizing properties). This class of maximal neighborhood searches has been shown to be important for finding perfect elimination orderings of chordal graphs and minimal triangulations of arbitrary graphs [1].

In summary we present our resulting hierarchy of simple vertex ordering characterizations in Fig. 1. Though we have concentrated on simple graphs, these ideas can easily be extended to more general environments. Our characterizations make no mention of multiple edges, but are equally applicable to searches of multigraphs. A search ordering implicitly imposes an orientation on the edges of a simple graph; it is straightforward to apply the same ideas to directed graphs to obtain similar characterizations. These ideas also hold for structures beyond graphs, such as for searches on hypergraphs. We refer to [6] for details.

Such characterizing properties can be a very powerful and useful tool for studying graphs. Beyond giving us a better understanding of how a particular search reveals the structure of a graph, it allows us to employ (and analyze) multiple sweeps of a search to gather additional information about a graph. We expect that this work is the first step to developing new multi-sweep algorithms for searches beyond just LexBFS.

References

[1] Berry, A., R. Krueger and G. Simonet, *Ultimate generalizations of LexBFS and LEX M*, in: *Proceedings of WG2005*, to appear.

[2] Brandstädt, A., F. Dragan and F. Nicolai, *LexBFS-orderings and powers of chordal graphs*, Discrete Math. **171** (1997), pp. 27–42.

[3] Corneil, D. G., *Lexicographic breadth first search — a survey*, in: *Proceedings of WG2004*, LNCS **3353**, 2004, pp. 1–19.

[4] Corneil, D. G. and R. Krueger, *A unified view of graph searching* (submitted).

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

[6] Krueger, R., "Graph Searching," Ph.D. thesis, University of Toronto (in preparation).

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

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