Graph Traversals as Universal Constructions

Siddharth Bhaskar
Department of Computer Science, University of Copenhagen, Denmark

Robin Kaarsgaard
School of Informatics, University of Edinburgh, UK

Abstract

We exploit a decomposition of graph traversals to give a novel characterization of depth-first *and* breadth-first traversals by means of universal constructions. Specifically, we introduce functors from two different categories of *edge-ordered* directed graphs into two different categories of transitively closed edge-ordered graphs; one defines the lexicographic depth-first traversal and the other the lexicographic breadth-first traversal. We show that each functor factors as a composition of universal constructions, and that the usual presentation of traversals as linear orders on vertices can be recovered with the addition of an inclusion functor. Finally, we raise the question of to what extent we can recover search algorithms from the categorical description of the traversal they compute.

2012 ACM Subject Classification Mathematics of computing → Paths and connectivity problems; Theory of computation → Models of computation

Keywords and phrases graph traversals, adjunctions, universal constructions, category theory

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.17

Related Version *Full Version*: `https://arxiv.org/abs/2104.14877` [4]

Funding *Siddharth Bhaskar*: Supported by *DFF | Natural Sciences* research project 1 fellowship 115554 *Cons-Free Recursion Theory.*
Robin Kaarsgaard: Supported by *DFF | Natural Sciences* international postdoctoral fellowship 0131-00025B *Landauer Meets von Neumann: Reversibility in Categorical Quantum Semantics.*

Acknowledgements We would like to thank Steve Lindell and Scott Weinstein for inspiring us to study traversals in terms of their predecessor functions, as well as suggesting the characterization of breadth-first traversals in Corollary 27. We also thank Jade Master for discussions relating to this work. We are indebted to the anonymous reviewers for their thorough and detailed comments.

1 Introduction

Graph searches are algorithms for visiting the vertices in a connected graph from a prescribed source. Both graph searches and their resulting vertex orders, or *traversals*, are absolutely fundamental in the theory of graph algorithms, and have important applications in other areas of theoretical computer science such as computational complexity theory.

We start from the premise that a concept as natural as a traversal should be obtainable canonically from the original graph. For example, let us consider the graph G in Figure 1, and fix a as a source. Of the six vertex orderings of G starting with a, two are not traversals: (a, d, b, c) and (a, d, c, b). This is because while searching a graph, each vertex added must be in the boundary of previously visited vertices, but d is not in the boundary of $\{a\}$.

Of the four remaining vertex orders, two are breadth-first traversals ((a, b, c, d) and (a, c, b, d)) and two are depth-first traversals ((a, b, d, c) and (a, c, d, b)). This can easily be checked by hand: in a breadth-first search, we go level-by-level, and must visit both b and c before we visit d. In a depth-first search, we prioritize the neighbor of the most recently visited vertex, so we visit d before the latter of $\{b, c\}$.

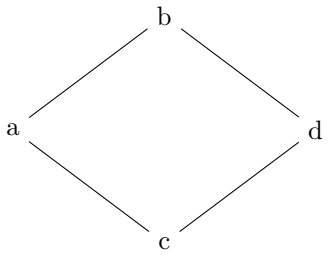


Figure 1 A 4-cycle.

However, notice that there is no way of canonically distinguishing between the two breadth-first or the two depth-first traversals. Concretely, once we visit a, there is no canonical way to choose between b and c. A natural fix is to linearly order each neighborhood and visit lesser neighbors first. We call the resulting traversals *lexicographic*. For example, if we say that $b < c$, then the lexicographic breadth-first traversal is (a, b, c, d) and the lexicographic depth-first traversal is (a, b, d, c). If we say that $c < b$, we get (a, c, b, d) and (a, c, d, b) respectively. We call a graph whose neighborhoods are linearly ordered an *edge-ordered* graph.

In the present paper, we show that both the lexicographic breadth-first traversal and lexicographic depth-first traversal are canonically obtainable from a given edge-ordered graph with a distinguished source. Specifically, we equip edge-ordered graphs with two different kinds of morphisms, and obtain lexicographic breadth- and depth-first traversals by applying a functor out of each category of edge-ordered graphs into the category of linear orders. We furthermore factor each functor as a composition of a forgetful functor and a sequence of *universal* (free and cofree) constructions on edge-ordered graphs.

At a first approximation, each lexicographic traversal can be expressed as the composition of a least-path tree and a transitive closure (see Figure 2). This decomposition was first observed in [7] – not in the context of category theory – where it was used to derive efficient parallel algorithms; see also [3, 6]. The main technical contribution of our paper is in identifying precisely the right notions of edge-ordered graphs and homomorphisms that allow us to formulate least-path trees and transitive closures as universal constructions.

To the best of our knowledge, equipping algorithmic problems with a categorical structure is a relatively recent idea. While graphs have been studied extensively from a categorical point of view, the focus has been on topics such as graph rewriting and string diagrams [8, 10] and relationships with properads [9] rather than graph algorithms. Closer to our approach, a surprisingly simple and elegant characterization of reachability in all *coalgebras* (i.e., including graphs) is studied in [2, 14]. A categorical treatment of the open algebraic path problem is given in [11], and a compositional algorithm for reachability in Petri nets is described in [12].

In [1], Abramsky describes a "great divide" between those areas of theoretical computer science focused on *structure* (semantics and type theory), and those focused on *power* (computability and complexity theory); they have "almost disjoint communities" with "no common technical language or tools." He proposes a high-level "theory-building" program of integrating these approaches with the intent of solving hard problems, akin to Grothendieck's program in algebraic geometry. We envision developing a theory of *compositional graph algorithms through universal properties* as a step along this way. A long-term goal of such a project would be to see whether one could recover *algorithms* from *problem statements*, if the latter are suitably formulated.

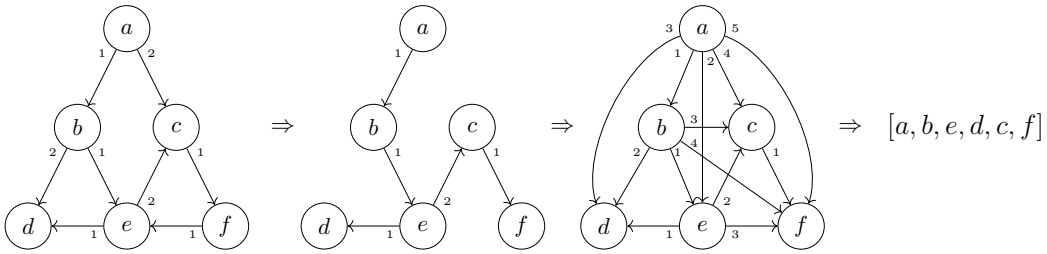


Figure 2 The construction of the lexicographic depth-first traversal starting from a on a given edge-ordered graph. First we extract the least-path tree, transitively close it, then isolate the ordered neighborhood of a. Numerals indicate the edge ordering. Each transformation is universal, except for a "silent" (forgetful) transformation fixing the transitive graph but forgetting some structure on morphisms.

This paper is structured as follows: We describe the necessary background on graphs and (lexicographic) traversals in Section 2, and present an alternate formulation of traversals in Section 3 that the remaining work will build on. We go on to describe the categories of edge-ordered graphs on which our work is founded in Section 4, and present the two categories of least path trees and universal constructions associated with lexicographic breadth-first and depth-first traversals respectively in Section 5 and Section 6. The categories of transitively closed least path trees and their universal constructions are presented in Section 7 and Section 8. Finally, we summarise our categorical construction of lexicographic breadth-first and depth-first traversals by universal means in Section 9, and end with some concluding remarks in Section 10.

Note that in the interests of space, we omit proofs in the pre-categorical development (Sections 2 and 3), and we relegate the category-theoretic proofs from Sections 4 through Section 9 to the Appendix. The omitted proofs may be found in [4].

2 Background and Preliminaries

Our objective is to give a categorical formulation of the canonical breadth-first and depth-first traversals of directed edge-ordered graphs. But first, in Sections 2 and 3, we give a "pre-categorical" account of these traversals in terms of extremal-path trees, which motivates the subsequent categorical treatment in Sections 4 through 8. The main results of Sections 2 and 3 can be found in [7], but we provide our own presentation.

▶ **Definition 1.** *A (directed) graph is a pair* $(V, \to)$ *where* V *is the set of* vertices *and* $\to \subseteq V \times V$ *is the* edge relation.

▶ **Definition 2.** *The* neighborhood *of a vertex* $v \in V$*, denoted* $N(v)$*, is the set of* outgoing *edges of* v*; that is,* $N(v) = \to \cap \{(v, x) \mid x \in V\}$*. A graph with a distinguished vertex is* pointed*.*

▶ **Definition 3.** *A* path *in a graph is a finite sequence of vertices* $v_1 \to v_2 \to \cdots \to v_n$*; say that* v_1 *is the* source *of the path, and* v_n *the* target*. We will write* $u \rightsquigarrow v$ *to say that there is a path from* u *to* v*, and* $u \overset{\pi}{\rightsquigarrow} v$ *about a specific such path named* π*. We say that a path is* proper *when no vertex occurs more than once in the path, i.e., when* $v_i = v_j$ *implies* $i = j$ *for all* $i, j \in I$*. We use* ε *to denote the unique* empty *(proper) path in each neighborhood.*

▶ Remark 4. Notice that $u \rightsquigarrow v$ iff there exists a proper path from u to v, as repetitions in a path can always be deleted.

▶ **Definition 5.** *Two paths are* co-initial *in case they share a source, and* co-final *in case they share a target. If the source of σ agrees with the target of π, we can* compose *them to obtain $\pi\sigma$. For two co-initial paths π and σ, we say $\pi \sqsubset \sigma$ (read: σ* extends *π) in case π is a proper initial subsequence (i.e., a proper prefix) of σ.*

▶ **Definition 6.** *A pointed graph (G, v_0) is* connected *if for every vertex v, there exists a path $v_0 \rightsquigarrow v$.*

2.1 Graph searching and traversals

By a *graph search*, we mean an algorithm for visiting all the vertices in a connected graph, starting at a given source. Two of the most important types of graph search are *depth-first* and *breadth-first* :

▶ **Definition 7** (Depth-first search)**.** *Given as input a finite connected graph $G = (V, E)$, initialize a list $L = ()$, and a stack $S = (v_0)$ for some vertex v_0. While S is nonempty, pop the first element v from S. If v is already contained in L, go back to the start of the loop. Otherwise, let $L = (L, v)$, and push every vertex in ∂v onto S, where $\partial v = N(v) \setminus L$.*

▶ **Definition 8** (Breadth-first search)**.** *Given as input a finite connected graph $G = (V, E)$, initialize a list $L = ()$, and a queue $Q = (v_0)$ for some vertex v_0. While S is nonempty, dequeue the front element v from Q. If v is already contained in L, go back to the start of the loop. Otherwise, let $L = (L, v)$, and enqueue every vertex in ∂v to the back of Q, where $\partial v = N(v) \setminus L$.*

Note that depth-first and breadth-first search are nondeterministic, in the sense that we do not specify which order to add vertices from ∂v in. Moreover, vertices may occur more than once in S or Q, as the same vertex may be added as a neighbor multiple times. However, vertices may not occur more than once in L; when depth-first or breadth-first search is complete, L lists all the vertices in G.

As we vary over all the nondeterministic traces of depth-first search or breadth-first search over a graph G, the resulting orderings are depth-first or breadth-first traversals.

▶ **Definition 9.** *A linear ordering $<$ of the vertices of a graph G is a* depth-first traversal*, respectively* breadth-first traversal *of G if there exists some computation of depth-first search, respectively breadth-first search, on G according to which vertices are added to L in exactly the order $<$.*

Both depth- and breadth-first traversals have important characterizations in the first-order language of ordered graphs; these are due to Corneil and Krueger [5], who state them in the special case of undirected graphs. Here, we only need to know that they are necessary.

▶ **Lemma 10.** *Suppose G is a finite, connected graph and $\cdot < \cdot$ is a depth-first traversal of G. Then for any vertices $u < v < w$ such that $u \to w$, there exists some v' such that $u \leq v' < v$ and $v' \to v$.*

▶ **Lemma 11.** *Suppose G is a finite, connected graph and $\cdot < \cdot$ is a breadth-first traversal of G. Then for any vertices $u < v < w$ such that $u \to w$, there exists some v' such that $v' \leq u < v$ and $v' \to v$.*

These characterizations motivate the following definitions.

▶ **Definition 12.** *Let G be a finite connected graph and $\cdot < \cdot$ be a depth-first traversal of G. Then for any vertex v, define the* depth-first predecessor *$dfp(v)$ to be the greatest $u < v$ such that $u \to v$, if v is not the minimal vertex. For the minimal vertex v, let $dfp(v) = v$.*

▶ **Definition 13.** *Let G be a finite connected graph and $\cdot < \cdot$ be a breadth-first traversal of G. Then for any vertex v, define the* breadth-first predecessor *$bfp(v)$ to be the least $u < v$ such that $u \to v$, if v is not the minimal vertex. For the minimal vertex v, let $bfp(v) = v$.*

Depth- and breadth-first predecessors allow for elegant restatements of Lemmata 10 and 11: if $\cdot < \cdot$ is a depth-first traversal of G, then for any $v < w$, if $dfp(w) < v$, then $dfp(w) \leq dfp(v)$. Similarly, if $\cdot < \cdot$ is a breadth-first traversal of G then for any $v < w$, if $bfp(w) < v$, then $bfp(v) \leq bfp(w)$. In fact the second condition is equivalent to saying that bfp is weakly monotone, viz., $v \leq w \implies bfp(v) \leq bfp(w)$.

For the next definition, notice that the orbits $\{v, dfp(v), dfp^2(v), \dots\}$ and $\{v, bfp(v), bfp^2(v), \dots\}$ of any vertex v are finite, and their *reversals* are paths from the least vertex v_0 to v.

▶ **Definition 14.** *Let G be a finite connected graph, $\cdot < \cdot$ be a depth-first traversal of G, and v_0 be the $<$-least vertex. Then the* canonical df-path *from v_0 to any vertex v is the path $v_0 \to v_1 \to \cdots \to v_{\ell-1}$ such that $v_{\ell-1} = v$, and for every $0 \leq i < \ell - 1$, $dfp(v_{i+1}) = v_i$.*

Analogously, the canonical bf-path *from v_0 to any vertex v is the path $v_0 \to v_1 \to \cdots \to v_{\ell-1}$ such that $v_{\ell-1} = v$, and for every $0 \leq i < \ell - 1$, $bfp(v_{i+1}) = v_i$.*

2.2 Lexicographic searching

The objective of this paper is to show that the depth-first and breadth-first traversals of a graph is *canonical* in some precise categorical sense. Of course, this cannot be true at face value: in a complete graph, every ordering of the vertices is both a breadth-first and a depth-first traversal, and none of them is canonical.

As we remarked in the introduction, an edge-ordering is precisely the amount of additional structure we need. Over such graphs, we can make searching deterministic, as we now show.

▶ **Definition 15.** *A* finite edge-ordered graph *is a finite graph where each neighborhood is equipped with a (strict) linear order $\cdot \triangleleft \cdot$.*

▶ **Definition 16** (Lexicographic depth-first search)**.** *Let G be a finite, pointed, connected, edge-ordered graph with distinguished vertex v_0. Initialize a list $L = ()$ and a stack $S = (v_0)$. While S is nonempty, pop the first element v from S. If v is already contained in L, go back to the start of the loop. Otherwise, let $L = (L, v)$, and push every vertex in ∂v onto S in* reverse *$\triangleleft$-order, where ∂v is the set of neighbors of v not in L.*

▶ Remark 17. We push vertices from ∂v onto S in reverse $\triangleleft$-order, so that the least vertices from ∂v end up on top of S.

▶ **Definition 18** (Lexicographic breadth-first search)**.** *Let G be a finite, pointed, connected, edge-ordered graph with distinguished vertex v_0. Initialize a list $L = ()$ and a queue $Q = (v_0)$. While Q is nonempty, dequeue first element v from S. If v is already contained in L, go back to the start of the loop. Otherwise, let $L = (L, v)$, and enqueue every vertex from ∂v onto Q in $\triangleleft$-order, where ∂v is the set of neighbors of v not in L.*

▶ **Definition 19.** *The depth-first traversal computed by lexicographic depth-first search on a finite, pointed, connected, edge-ordered graph G, is its* lexicographic depth-first traversal*. Similarly, the breadth-first traversal computed by lexicographic breadth-first search is its* lexicographic breadth-first traversal*.*

▶ Remark 20. The usage of *lexicographic (depth, breadth)-first (search, traversal)* is ambiguous in the literature. Here, we use it in the same way as Delatorre and Kruskal [7, 6]; however, the *lexicographic breadth-first search* of Rose and Tarjan [13] and the analogous *depth-first* version of Corneil and Krueger [5] are different. The latter are further refinements of breadth-first search and depth-first search over graphs, not a "determinization" by an edge-ordering.

3 Orders on paths

In this section, we give different characterizations of the lexicographic depth-first and breadth-first traversals, independent of any graph search, which suggest the category-theoretic treatment that occupies us for the rest of the paper.

▶ **Definition 21.** *Fix an edge-ordered graph G. Define the* lexicographic path relation $\cdot \prec \cdot$ *on any proper co-initial paths π and σ in G as follows:*

(i) *if $\pi \sqsubset \sigma$, then $\pi \prec \sigma$, similarly if $\sigma \sqsubset \pi$, then $\sigma \prec \pi$; otherwise,*

(ii) *let ζ be the longest common prefix of π and σ, let u be the target of ζ, and let v_1 and v_2 be the first vertices immediately following ζ in π and σ respectively. Order $\pi \prec \sigma \iff u \to v_1 \lhd u \to v_2$.*

▶ Remark 22. The following properties hold of $\prec$:

1. The empty path (v) is least among all paths from v.
2. If $\pi \prec \sigma$ and $\pi \not\sqsubseteq \sigma$, then for any α and β, $\pi\alpha \prec \sigma\beta$.
3. If $\pi \prec \sigma$, then $\alpha\pi \prec \alpha\sigma$.
4. If $\alpha\pi \prec \alpha\sigma$, then $\pi \prec \sigma$.

Since the set of proper paths is finite, any subset has a $\prec$-least element, which justifies the next definitions.

▶ **Definition 23.** *For vertices u, v in a finite edge-ordered graph, let $\min(u \rightsquigarrow v)$ be the lexicographically least proper path from u to v.*

▶ **Definition 24.** *For vertices u, v in a finite edge-ordered graph, let $\min^s(u \rightsquigarrow v)$ be the lexicographically least* shortest *path from u to v.*

In fact, it will be convenient to define the following relation, the *shortlex* order:

▶ **Definition 25.** *Let $\pi \prec^s \sigma$ mean that either $|\pi| < |\sigma|$, or $|\pi| = |\sigma|$ and $\pi \prec \sigma$.*

In this case, $\min^s(u \rightsquigarrow v)$ is simply, the $\prec^s$-least path $u \rightsquigarrow v$.

In the remainder of this section, we fix a finite, pointed, connected, edge-ordered graph G with distinguished element v_0. Reserve the symbol $\cdot \lhd \cdot$ for the given ordering on co-initial edges and $\cdot \prec \cdot$ for the induced lexicographic ordering on co-initial paths. Let $\cdot <_D \cdot$ and $\cdot <_B \cdot$ denote the lexicographic depth-first and breadth-first traversals respectively. Let $\mathfrak{P}v$ and $\mathfrak{Q}v$ denote the canonical df- and bf-paths from v_0 to v with respect to $\cdot <_D \cdot$ and $\cdot <_B \cdot$ respectively.

Our goal is to prove that $<_D$ and $<_B$ satisfy the following properties,

$$u <_D v \iff \mathfrak{P}u \prec \mathfrak{P}v$$

$$u <_B v \iff \mathfrak{Q}u \prec^s \mathfrak{Q}v,$$

which then in turn yield the following alternate characterizations:

$$u <_D v \iff \min(v_0 \rightsquigarrow v) \prec \min(v_0 \rightsquigarrow u)$$

$$u <_B v \iff \min^s(v_0 \rightsquigarrow v) \prec \min^s(v_0 \rightsquigarrow u).$$

▶ **Lemma 26.** *If $u \leq_B v$, then for any path $\pi : v_0 \rightsquigarrow v$, $|\mathfrak{Q}u| \leq |\pi|$. In addition, if $|\mathfrak{Q}u| = |\pi|$, then $\mathfrak{Q}u \preceq \pi$.*

▶ **Corollary 27.** *The following are all true of the operator* $\mathfrak{Q}$*:*
1. *For any vertex* v *of* G, $\mathfrak{Q}v = \min^s(v_0 \rightsquigarrow v)$.
2. $u \leq_B v$ *iff* $\mathfrak{Q}u \preceq^s \mathfrak{Q}v$.

Corollary 27 yields $u <_B v \iff \min^s(v_0 \rightsquigarrow v) \prec \min^s(v_0 \rightsquigarrow u)$, as desired.

▶ **Definition 28.** *Define* $v \prec_D w$ *in case* $\min(v_0 \rightsquigarrow v) \prec \min(v_0 \rightsquigarrow w)$.

▶ **Lemma 29.** *For any vertex* v *of* G, $\mathfrak{P}v = \min(v_0 \rightsquigarrow v)$.

▶ **Theorem 30.** *The order* $\cdot \prec_D \cdot$ *is exactly the lexicographic depth-first traversal* $\cdot <_D \cdot$ *of* G.

4 Categories of graphs

We now work towards a categorical formulation of the above material. This means we need to categorify, i.e., equip with morphisms, all the objects we defined in Section 2, as well as introduce some new objects.

Continuing the convention established above, we use they symbol $\prec$ and variations thereof to refer to orderings of co-initial paths, and the symbol $\lhd$ to refer to neighborhood orders, i.e., orderings of co-initial edges. We reserve the symbol $<$ for vertex orders, but these will not reappear until Section 9.

▶ **Definition 31.** *A* homomorphism of graphs $G \xrightarrow{h} H$ *is a function from* G*-vertices to* H*-vertices which preserves edge connectivity: For all* G*-vertices* u, v, $u \to v$ *in* G *implies* $h(u) \to h(v)$ *in* H*. A* homomorphism of pointed graphs $G \xrightarrow{h} H$ *must additionally map the distinguished vertex of* G*, and* only *that vertex, to the distinguished vertex of* H*.*

▶ **Definition 32.** *If* $h : G \to H$ *is a graph homomorphism and* π *is the path* $v_1 \to v_2 \to \ldots v_n$ *in* G*, then* $h(\pi)$ *is defined to be the path* $h(v_1) \to h(v_2) \to \ldots h(v_n)$ *in* H*.*

▶ Remark 33. The following hold of any homomorphism $h : G \to H$;
1. $h(\varepsilon) = \varepsilon$, $h(\pi\sigma) = h(\pi)h(\sigma)$, and if $\pi \sqsubset \sigma$, then $h(\pi) \sqsubset h(\sigma)$.
2. If h is injective on vertices and π is a proper path, then $h(\pi)$ is a proper path.
3. If h is injective on vertices, then h preserves longest common prefixes.

We now define homomorphisms of edge-ordered graphs (cf. Definition 15). In addition to the "straightforward" property of being monotone on neighborhoods, we also want to consider homomorphisms that preserve lexicographically least paths. This gives us two refinements of the notion of homomorphism, depending on whether we want to preserve lexicographically ($\prec$) least paths or short-lex ($\prec^s$) least paths.

▶ **Definition 34.** *A* homomorphism *of finite, pointed, edge-ordered graphs* $G \xrightarrow{h} H$ *is a homomorphism of pointed graphs that is monotone on neighborhoods; explicitly,*

(i) $u \to v$ *in* G *implies* $h(u) \to h(v)$ *in* H*,*

(ii) v_G *is the* unique *vertex such that* $h(v_G) = v_H$*, where* v_G *and* v_H *are the distinguished points of* G *and* H *respectively, and*

(iii) $u \to v_1 \lhd u \to v_2$ *implies* $h(u) \to h(v_1) \lhd h(u) \to h(v_2)$.

In addition, a lex-homomorphism *must preserve least paths:*

(iv) $h(\min(u \rightsquigarrow v)) = \min(h(u) \rightsquigarrow h(v))$,

and a short-lex homomorphism *must preserve least shortest paths:*

(v) $h(\min^s(u \rightsquigarrow v)) = \min^s(h(u) \rightsquigarrow h(v))$.

▶ **Lemma 35.** *If* $h : G \to H$ *is a homomorphism of edge-ordered graphs then* h *preserves* $\cdot \prec \cdot$ *as well as* $\cdot \prec^s \cdot$.

Table 1 Categories of edge-ordered graphs and their morphisms.

Category	Objects	Morphisms
$\mathbf{FinGraph}^{<}_{\star}$	finite, pointed, connected edge-ordered graphs	pointed, edge-ordered graph homomorphisms
$\mathbf{FinGraph}^{l}_{\star}$	finite, pointed, connected edge-ordered graphs	lex-homomorphisms *(pointed, edge-ordered graph homomorphisms that preserve least paths)*
$\mathbf{FinGraph}^{s}_{\star}$	finite, pointed, connected edge-ordered graphs	short-lex homomorphisms *(pointed, edge-ordered graph homomorphisms that preserve least shortest paths)*
LexGraph	lex-graphs *(finite, pointed, connected, and edge-ordered by definition)*	lex-homomorphisms
$\mathbf{FinArb}^{<}_{\star}$	finite, pointed, edge-ordered arborescences	pointed, edge-ordered graph homomorphisms
TLexGraph	transitive lex-graphs	lex-homomorphisms

An important special case of edge-ordered graphs are those where the edge order agrees with the path order in each neighborhood.

▶ **Definition 36.** *A* lex-graph *is a finite, pointed, connected, edge-ordered graph such that the edge order is compatible with the lexicographic path order; explicitly, it is a finite directed graph equipped with*

(i) *a distinguished vertex v_0, and*

(ii) *a linear order $\lhd$ on each neighbourhood $N(u)$ such that for every $v_1, v_2 \in N(u)$, $\min(u \rightsquigarrow v_1) \prec \min(u \rightsquigarrow v_2) \iff u \to v_1 \lhd u \to v_2$.*

▶ Remark 37. We might be tempted to define, analogously, a *short-lex graph* by demanding that $v_1, v_2 \in N(u)$, $\min^s(u \rightsquigarrow v_1) \prec^s \min^s(u \rightsquigarrow v_2) \iff u \to v_1 \lhd u \to v_2$. However notice that this condition simply holds automatically for any edge-ordered graph: if there is an edge $u \to v$, that edge is the lexicographically least shortest path.

Finally, we isolate two extremal special cases of lex-graphs, one with very few edges, and one with very many.

▶ **Definition 38.** *An* arborescence *is a pointed directed graph $G = (V, \to, v_0)$ such that for every vertex u, there is a unique path $v_0 \rightsquigarrow u$ in G.*

▶ Remark 39. If $(V, \to, v_0)$ is an arborescence, its underlying undirected graph (V, E) is connected and acyclic (where $E(u, v)$ iff $(u \to v)$ or $(v \to u)$), and every edge $u \to v$ is oriented away from v_0, meaning that the distance from v_0 to u is less than the distance from v_0 to v. If G is a finite edge-ordered graph that is also an arborescence, then it is already lex-graph, since the only path between u and $v \in N(u)$ *is* the edge $u \to v$. Moreover, if S and T are arborescences and if $h : S \to T$ is a pointed, edge-ordered graph homomorphism, then it is already both a lex-homomorphism and a short-lex homomorphism.

▶ **Definition 40.** *A graph is* transitive *if its edge relation is; i.e., if $u \to v$ and $v \to w$ implies $u \to w$. A* transitive lex-graph *is a lex-graph with a transitive edge relation.*

It follows readily that this zoo of graph variations each form a category, summarized in Table 1. Diagrammatically, we show the relationships between these categories in Figure 3, where each arrow indicates inclusion as a subcategory.

There is one additional category that will be introduced in Section 8, and that is **TArb** (Definition 51) whose objects are transitive closures of arborescences with a particular edge order depending on the underlying arborescence. Curiously, their morphisms preserve *longest* paths instead of shortest paths.

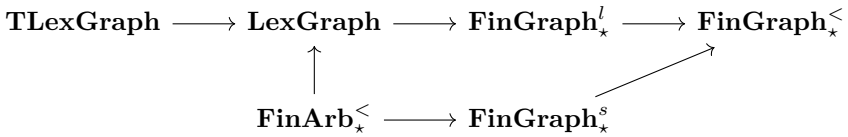


Figure 3 Categories of edge-ordered graphs and their relationships.

5 Least path trees

Given a finite, pointed, connected, edge-ordered graph, we can delete all edges which are not on some lexicographically least path starting from the distinguished vertex v_0. We get a finite, edge-ordered arborescence. In this section, we show that this construction (denoted Θ) is cofree, indeed coreflective: it is right adjoint to the inclusion functor I.

▶ **Definition 41** (The functor Θ). *Given a finite, pointed, connected, edge-ordered graph G, with distinguished vertex v_0, the graph $\Theta(G)$ is defined as follows:*

- *the vertices of $\Theta(G)$ are the vertices of G, and*
- *any edge $u \to v$ appears in $\Theta(G)$ iff $u \to v$ is contained in $\min(v_0 \rightsquigarrow v)$.*
- *Order co-initial edges $u \to v_1 \triangleleft u \to v_2$ in $\Theta(G)$ iff the same relation holds in G.*

For a lex-homomorphism $h : G \to H$ define $\Theta(h) : \Theta(G) \to \Theta(H)$ by $\Theta(h)(v) = h(v)$ for any vertex $v \in \Theta(G)$.

The proof that Θ is a well-defined functor into the indicated category is in the appendix (Lemma 61). Next we define I, which is simply inclusion of categories.

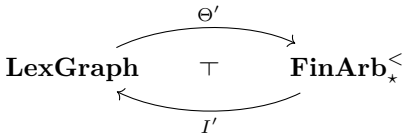
▶ **Definition 42** (The functor I). *Given a finite, edge-ordered arborescence T, the object $I(T)$ is simply identified with T. Given a pointed, edge-ordered homomorphism $h : S \to T$, the homomorphism $I(h) : I(S) \to I(T)$ is simply identified with h.*

Since every finite, edge-ordered arborescence is a finite, pointed, connected, edge-ordered graph, and since morphisms in $\mathbf{FinArb}^<_\star$ are defined to be morphisms of pointed, edge-ordered graphs, I is well-defined.

▶ **Theorem 43.** *There is an adjunction* $\mathbf{FinGraph}^l_\star \underset{I}{\overset{\Theta}{\top}} \mathbf{FinArb}^<_\star$.

Something for free

Since the image of I is always a lex-graph, and since **LexGraph** is a full subcategory of $\mathbf{FinGraph}^l_\star$, we actually get the following adjunction for free:



where I' is the functor I but with target category **LexGraph**, and Θ' is Θ restricted to **LexGraph**. The proof is in the appendix (Lemma 62).

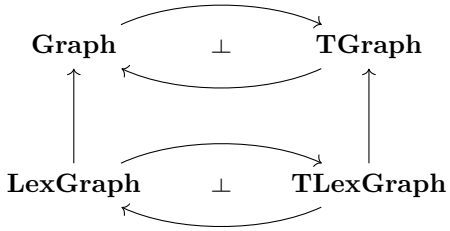


Figure 4 The lexicographic-transitive closure of lex-graphs specializes the transitive closure of finite graphs.

6 Least shortest-path trees

Our objective in this section is to establish a shortest-paths analogue of the previous section, i.e., to define a lexicographically least *shortest*-path tree functor $\mathbf{FinGraph}^s_\star \xrightarrow{S} \mathbf{FinArb}^<_\star$ and an inclusion functor $\mathbf{FinArb}^<_\star \xrightarrow{I} \mathbf{FinGraph}^s_\star$. (It is not, of course, the same I as in the previous section, the source category being different.)

▶ **Definition 44** (The functor S). *Given a finite, pointed, connected, edge-ordered graph G, with distinguished vertex v_0, the graph $S(G)$ is defined as follows:*

- *the vertices of $S(G)$ are identified with the vertices of G, and*
- *any edge $u \to v$ appears in $S(G)$ iff $u \to v$ is contained in $\min^s(v_0 \rightsquigarrow v)$.*
- *Order co-initial edges $u \to v_1 \lhd u \to v_2$ in $S(G)$ iff the same relation holds in G.*

If $h : G \to H$ is a homomorphism, define $S(h) : S(G) \to S(H)$ by $S(h)(v) = h(v)$, for any vertex $v \in S(G)$.

To show that S is functorial, it suffices to check that $S(G)$ is always an arborescence. We prove this in the appendix (Lemma 63).

▶ **Theorem 45.** *There is an adjunction* $\mathbf{FinGraph}^s_\star \underset{I}{\overset{S}{\rightleftarrows}} \mathbf{FinArb}^<_\star$ ($\top$).

7 Transitive closure of lex-graphs

There is a well-known adjunction between the category of graphs and the category **TGraph** of transitive graphs, where the functor in one direction transitively closes the edge relation, and in the other direction is simply inclusion [11]. Here, we refine this to an adjunction between **LexGraph** and **TLexGraph**. Indeed, the usual transitive closure appears when forgetting the order as in Figure 4 (where the functors (**T**)**LexGraph** $\to$ (**T**)**Graph** simply forget the edge order).

While it is immediately clear that there is a forgetful (inclusion) functor $\mathbf{TLexGraph} \xrightarrow{U} \mathbf{LexGraph}$, we must construct the free functor in the other direction.

▶ **Definition 46** (The functor F). *Given a lex-graph G, define the transitive lex-graph $F(G)$ as follows:*

- *the vertices of $F(G)$ are the vertices of G,*
- *the distinguished point of $F(G)$ is the distinguished point of G,*
- *The edge relation of $F(G)$ is the transitive closure of the edge relation of G, and finally*

- *for any vertices v_1, v_2 in a common neighborhood $N(u)$, let $u \to v_1 \lhd u \to v_2$ in $F(G)$ iff $\min(u \rightsquigarrow v_1) \prec \min(u \rightsquigarrow v_2)$ in G.*

On morphisms, given $G \xrightarrow{h} H$ we define $F(G) \xrightarrow{F(h)} F(H)$ by $F(h)(v) = h(v)$.

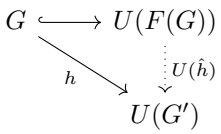
We must show that F is a well-defined functor, but before doing so, we state several important relationships between the lex-graph G and the edge-ordered graph $F(G)$:

▶ **Lemma 47.** *For any lex-graph G,*
- **(i)** *The neighborhood order of $F(G)$ extends that of G*
- **(ii)** *The lexicographic path order of $F(G)$ extends that of G.*
- **(iii)** *Least paths in G and $F(G)$ coincide.*

▶ **Lemma 48.** *The map* **LexGraph** $\xrightarrow{F}$ **TLexGraph** *is well-defined and functorial.*

▶ **Theorem 49.** *There is an adjunction* $\mathbf{LexGraph} \underset{U}{\overset{F}{\perp}} \mathbf{TLexGraph}$.

▶ Remark 50. In the title of this paper, we promise universal constructions – and while such do arise from adjunctions (from the unit and counit), it seems fitting that we make this explicit at least once. As a consequence of Theorem 49, for any lex-graph G in with lexicographic-transitive closure $U(F(G))$, given any other transitive lex-graph $U(G')$ and $G \xrightarrow{h} U(G')$ there is a *unique* homomorphism of transitive lex-graphs $F(G) \xrightarrow{\hat{h}} G'$ such that the diagram below commutes in **LexGraph**:



The reader is invited to extract similar universal mapping properties from other adjunctions in this paper.

8 Transitive closure of least shortest path trees

We now establish the final adjunction of our paper, and the second adjunction needed to characterize breadth-first traversals. Analogously to the depth-first case, this adjunction relates $\mathbf{FinArb}^{<}_{\star}$ and a category of transitively closed graphs:

▶ **Definition 51** (The category **TArb**). *A finite, pointed, connected, edge-ordered graph G is an object of* **TArb** *iff there exists a finite, pointed, connected, edge-ordered arborescence T and an identification of the vertices of G with the vertices of T such that*
- *the distinguished points of each are identical,*
- *the edge relation of G is the transitive closure of the edge relation of T, and*
- *$u \to v_1 \lhd u \to v_2$ in G iff $v_0 \rightsquigarrow v_1 \prec^s v_0 \rightsquigarrow v_2$ in T, where $v_0 \rightsquigarrow v$ is the* unique *path from v_0 to v in T.*

A map $G_1 \xrightarrow{h} G_2$ is a morphism in **TArb** *if it is a homomorphism of edge-ordered graphs ((i)–(iii) of Defintion 34) and in addition it preserves* longest *paths.*

This last property only makes sense if longest paths are unique. Luckily, longest paths in the transitive closure of an arborescence are exactly the edges of the original tree.

▶ **Lemma 52.** *If T is an arborescence (not necessarily edge-ordered) with root v_0, and G is its transitive closure, then an edge $u \to v$ of G is an edge of T iff $u \to v$ appears on a longest path $v_0 \rightsquigarrow v$ in G. Moreover, the longest path $u \rightsquigarrow v$ in G is unique.*

▶ Remark 53. As a consequence of Lemma 52, for any graph $G \in \mathbf{TArb}$, $u \to v_1 \triangleleft u \to v_2$ in G iff $v_0 \rightsquigarrow v_1 \prec^s v_0 \rightsquigarrow v_2$ in G, where $v_0 \rightsquigarrow v$ is the unique longest path from v_0 to v in G.

While the presentation of depth- and breadth-first traversals has thus far been similar even at a rather small scale, this adjunction diverges from the previous one in several ways: it does not decompose into an inclusion and a functor which lifts the ordinary transitive closure, and the curious preservation of longest paths by morphisms is not suggested by the pre-categorical treatment of breadth-first traversals. This category arises somewhat mysteriously, and we have no justification for introducing it other than it works.

We now define the two functors of the adjunction. The proofs that they are functorial are Lemmas 64 and 65 in the appendix.

▶ **Definition 54.** *Define the functor $\mathbf{FinArb}^{<}_{\star} \xrightarrow{\Gamma} \mathbf{TArb}$ by $\Gamma(V, v_0, \to) = (V, v_0, \xrightarrow{trans})$, and for any vertices (u, v_1, v_2) such that $u \to v_1$ and $u \to v_2$ in $\Gamma(T)$, define $u \to v_1 \triangleleft u \to v_2$ iff $v_0 \rightsquigarrow v_1 \prec^s v_0 \rightsquigarrow v_2$ in T. For morphisms $T \xrightarrow{h} T'$, define $\Gamma(h) : \Gamma(T) \to \Gamma(T')$ by $\Gamma(h)(v) = h(v)$.*

▶ **Definition 55.** *Define the functor $\mathbf{TArb} \xrightarrow{\mathrm{L}} \mathbf{FinArb}^{<}_{\star}$ by identifying the set of vertices of $\mathrm{L}(G)$ with the vertices of G, and including an edge $u \to v$ in $\mathrm{L}(G)$ iff it lies on the longest path $v_0 \rightsquigarrow v$ in G. The neighborhood order in $\mathrm{L}(G)$ is inherited from G, and for morphisms $G \xrightarrow{h} H$, define $\mathrm{L}(h) : \mathrm{L}(G) \to \mathrm{L}(H)$ by $\mathrm{L}(h)(v) = h(v)$.*

▶ **Theorem 56.** *There is an adjunction* $\mathbf{FinArb}^{<}_{\star} \underset{\mathrm{L}}{\overset{\Gamma}{\rightleftarrows}} \mathbf{TArb}$.

9 Putting it all together

Combining the adjunctions from Sections 5 and 7, we get a chain of adjunctions

$$\mathbf{FinGraph}^{<}_{*} \underset{I}{\overset{\Theta}{\rightleftarrows}} \mathbf{FinArb}^{<}_{*} \underset{\Theta'}{\overset{I'}{\rightleftarrows}} \mathbf{LexGraph} \underset{U}{\overset{F}{\rightleftarrows}} \mathbf{TLexGraph}$$

Since adjunctions of the same handedness compose, we can rewrite this as

$$\mathbf{FinGraph}^{<}_{*} \underset{I}{\overset{\Theta}{\rightleftarrows}} \mathbf{FinArb}^{<}_{*} \underset{\Theta' \circ U}{\overset{I' \circ F}{\rightleftarrows}} \mathbf{TLexGraph}$$

Similarly, by combining the adjunctions of Sections 6 and 8 we get

$$\mathbf{FinGraph}^{s}_{*} \underset{I}{\overset{S}{\rightleftarrows}} \mathbf{FinArb}^{<}_{*} \underset{L}{\overset{\Gamma}{\rightleftarrows}} \mathbf{TArb}$$

Suppose we fix a finite, pointed, edge-ordered graph, locate it in either $\mathbf{FinGraph}^l_\star$ or $\mathbf{FinGraph}^s_\star$, then apply the appropriate least-path tree and the transitive closure functors. Then the edge ordering of the distinguished vertex in the result is the lexicographic depth-first or breadth-first traversal respectively.

▶ **Lemma 57.** *Given any $G \in \mathbf{FinGraph}^l_\star$, let $T = (F \circ I' \circ \Theta)(G)$. Then the neighborhood ordering $\lhd$ on the distinguished point in T is the lexicographic depth-first traversal of G.*

▶ **Lemma 58.** *Given any $G \in \mathbf{FinGraph}^s_\star$, let $T = (\Gamma \circ S)(G)$. Then the neighborhood ordering $\lhd$ on the distinguished point in T is the lexicographic breadth-first traversal of G.*

Finally, we show that these traversals can be extracted as vertex orders rather than edge orders, by defining a functor that takes an edge-ordered graph into the ordered neighborhood of its distinguished point. Let **FinLoset** denote the category of finite linearly ordered sets with monotone functions.

▶ **Lemma 59.** *There is a functor $E : \mathbf{FinGraph}^<_\star \to \mathbf{FinLoset}$ that linearly orders the vertices in a given graph according to the ordering on $N(v_0)$.*

Since there are inclusions $\mathbf{TLexGraph} \to \mathbf{FinGraph}^<_\star$ and $\mathbf{TArb} \to \mathbf{FinGraph}^<_\star$, we get

▶ **Corollary 60.** *There are functors $\mathbf{FinGraph}^l_\star \to \mathbf{FinLoset}$ and $\mathbf{FinGraph}^s_\star \to \mathbf{FinLoset}$ which compute the lexicographic depth-first and breadth-first traversals respectively.*

Note that parts of this final step $\mathbf{FinGraph}^<_\star \to \mathbf{FinLoset}$ can also be made universal, but since we still have to accept the presence of inclusion functors without adjoints, we only sketch this construction: The functor E from Lemma 59 restricts to a functor from the category of *transitive* finite, pointed, edge-ordered graphs and their homomorphisms, and into the category of finite *nonempty* linearly ordered sets with monotone functions *which additionally preserve the least element*. This restricted functor has a left adjoint given by mapping a finite linearly ordered set $(V, <)$ to the graph with vertices V, distinguished vertex the least element v_0 of V (guaranteed to exist when V is finite), and edge relation given by $v_0 \to v$ for all $v \in V$ with $v \neq v_0$. This specializes to neither **TLexGraph** nor **TArb**, however, since the counit of this adjunction need not preserve either least or longest paths.

10 Discussion and open questions

We have described a construction of depth-first traversals from a category of finite edge-ordered graphs whose morphisms preserve least paths as a series of universal constructions, and in analogous fashion, a construction of breadth-first traversals from a category of finite edge-ordered graphs whose morphisms preserve least shortest paths. A diagram summarizing the various adjunctions involved is shown in Figure 5. The prevalence of coreflections in our work is nicely aligned with previous work on reachability in coalgebras [2, 14], where well-pointed and reachable coalgebras respectively turn out to form coreflective subcategories.

This begs the question of whether there is way of generalizing these two constructions using a general notion of extremal path. Such a method is suggested by Delatorre and Kruskal [7] using the machinery of a closed semiring system, which provides a general setting for several lexicographic algebraic path problems; not only breadth-first and depth-first search, but *lexicographic topological search* as well.

The deeper question raised by our work is how, if at all, the categorification of a problem informs the algorithms which solve it. For example, it seems natural that problems which can be expressed by a *single* (free or cofree) universal construction would be amenable to a

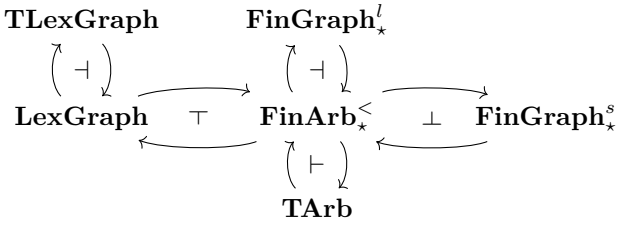


Figure 5 Categories of edge-ordered graphs and adjunctions between them.

solution by a greedy algorithm. Intuitively, if an object is free, then it can be built up *over here* without affecting what happens *over there*, and can thus be constructed by making local choices – exactly what a greedy algorithm does.

What if a problem can be expressed by a composition of two adjunctions of opposite handedness, as we show here? Is there an algorithmic strategy for such problems? More generally, is there a robust hierarchy of problems on graphs which classifies them by the number of free-cofree alternations, and can we unite problems in the same level of the hierarchy with a common algorithmic template?

Speculatively, we envision a world in which properties of algorithms could be inferred from the appropriate categorical formulation of the problems they solve. In such a world, a statement like *problem X can be solved using two sequential applications of depth-first search, but no fewer* would have a precise meaning. The semantics functor that transforms an algorithm into a problem that it solves could be decomposed in several meaningful ways, which would give different pieces of information about the algorithm. (Perhaps, for example, one decomposition would give upper bounds on the sequential complexity of solving the problem, and another on the parallel complexity.)

This is a long-term goal. In the medium term, there are still many compelling questions we might hope to answer. For example, the common sequential algorithm implementing depth-first search uses stacks, and the analogous algorithm for breadth-first search uses queues. Can we somehow infer "stacks" and "queues" from a common categorical formulation of depth-first and breadth-first traversals? Can we recover the parallel complexities of these problems proven in [6]?

Positive answers to these or similar questions would be strong evidence that the "categorical structure theory of algorithms" mentioned in the introduction is actually a viable program; that it can not only describe problems, but suggest ways to solve them – in the elegant characterization of [1], to *lead* rather than to *follow*.

References

1 S. Abramsky. Whither semantics? *Theoretical Computer Science*, 807:3–14, 2020.

2 J. Adámek, S. Milius, L. S. Moss, and L. Sousa. Well-pointed coalgebras. *Logical Methods in Computer Science*, 9(3), 2013.

3 E. Allender, A. Chauhan, and S. Datta. Depth-first search in directed graphs, revisited. *Electronic colloquium on computational complexity*, 20(74), 2020.

4 S. Bhaskar and R. Kaarsgaard. Graph traversals as universal constructions, 2021. `arXiv:2104.14877`.

5 D. Corneil and R. Krueger. A unified view of graph searching. *SIAM J. Discrete Math.*, 22:1259–1276, January 2008.

6 P. Delatorre and C. P. Kruskal. Polynomially improved efficiency for fast parallel single-source lexicographic depth-first search, breadth-first search, and topological-first search. *Theory of Computing Systems*, 34:275–298, 2001.

7 P. Delatorre and C.P. Kruskal. Fast parallel algorithms for all-sources lexicographic search and path-algebra problems. *Journal of Algorithms*, 19(1):1–24, 1995.

8 L. Dixon and A. Kissinger. Open-graphs and monoidal theories. *Mathematical Structures in Computer Science*, 23(2):308–359, 2013.

9 J. Kock. Graphs, hypergraphs, and properads. *Collectanea mathematica*, 67(2):155–190, 2016.

10 S. Lack and P. Sobociński. Adhesive and quasiadhesive categories. *RAIRO-Theoretical Informatics and Applications*, 39(3):511–545, 2005.

11 J. Master. The open algebraic path problem, 2020. `arXiv:2005.06682`.

12 J. Rathke, P. Sobociński, and O. Stephens. Compositional reachability in petri nets. In J. Ouaknine, I. Potapov, and J. Worrell, editors, *Reachability Problems*, pages 230–243. Springer, 2014.

13 D. J. Rose and R. E. Tarjan. Algorithmic aspects of vertex elimination. In *Proceedings of the Seventh Annual ACM Symposium on Theory of Computing (STOC '75)*, pages 245–254. ACM, 1975.

14 T. Wißmann, S. Milius, S. Katsumata, and J. Dubut. A coalgebraic view on reachability. *Commentationes Mathematicae Universitatis Carolinae*, 60(4):605–638, 2019.

A Appendix: Proofs from Sections 4 through 9

Proof of Lemma 35. It suffices to show that h preserves $\cdot \prec \cdot$; the second statement follows from the observation that graph homomorphisms preserve path length, i.e., $|\pi| = |h(\pi)|$ for every path π.

Suppose that π and σ are co-initial paths in G such that $\pi \prec \sigma$. If $\pi \sqsubset \sigma$, then $h(\pi) \sqsubset h(\sigma)$ and we're done. Otherwise, let $u \to v_1$ and $u \to v_2$ be the first edges in π and σ respectively following their longest common prefix ζ. Since $\pi \prec \sigma$, $u \to v_1 \lhd u \to v_2$, so $h(u) \to h(v_1) \lhd h(u) \to h(v_2)$.

Notice that $h(\zeta)$ is a common prefix of $h(\pi)$ and $h(\sigma)$, and since $h(u) \to h(v_1) \lhd h(u) \to h(v_2)$, it must be the longest one. Moreover, since $h(u) \to h(v_1) \lhd h(u) \to h(v_2)$, $h(\pi) \prec h(\sigma)$, which is what we wanted to show. ◀

▶ **Lemma 61.** *The functor Θ is well-defined.*

Proof. We first have to check that $\Theta(G)$ is a finite, edge-ordered arborescence. It is trivially finite and edge-ordered. Notice that $\Theta(G)$ is connected: for every vertex u, every edge on the path $\min(v_0 \rightsquigarrow u)$ is contained in $\Theta(G)$. On the other hand, the in-degree of each vertex is at most 1: there cannot be two distinct edges $u \to v$ and $u' \to v$ in $\min(v_0 \rightsquigarrow v)$.

Next we have to check that for any homomorphism h, $\Theta(h)$ is in fact a homomorphism of pointed, edge-ordered graphs. First notice that $\Theta(h)$ actually maps into $\Theta(H)$: if $u \to v$ is included in $\Theta(G)$, then it must be contained in $\min(v_0 \rightsquigarrow v)$ in G, whose h-image is $\min(h(v_0) \rightsquigarrow h(v))$ by property (iv) of Definition 34; therefore, $h(u \to v)$ is contained in a least path and so included in $\Theta(H)$.

Moreover, $\Theta(h)$ clearly fixes the distinguished vertex, and visibly inherits monotonicity on co-initial edges from h. Hence, $\Theta(h)$ is a pointed, edge-ordered graph homomorphism. ◀

Proof of Theorem 43. First, given $I(T) \xrightarrow{h^\uparrow} G$ in $\mathbf{FinGraph}^l_\star$, we describe how to obtain $T \xrightarrow{h_\downarrow} \Theta(G)$ in $\mathbf{FinArb}^<_\star$. Define $h_\downarrow(v) = h^\uparrow(v)$, for $v \in T$ (which, remember is the same as $I(T)$). To check that $h_\downarrow$ is well-defined, we must show that if $u \to v$ is an edge of T, then

$h^\uparrow(u \to v)$ is contained in $\Theta(G)$. But since T is an arborescence, $u \to v$ is trivially contained in $\min(v_0 \rightsquigarrow v)$, so $h^\uparrow(u \to v)$ is contained in $\min(h(v_0) \rightsquigarrow h(v))$, and hence included in $\Theta(G)$.

Moreover, the fact that $h_\downarrow$ maps edges to edges, preserves the distinguished point, and is monotone on co-initial edges, is immediately inherited from $h^\uparrow$.

Next, given $T \xrightarrow{h^\downarrow} \Theta(G)$ in $\mathbf{FinArb}^<_\star$, define $I(T) \xrightarrow{h^\uparrow} G$ in $\mathbf{FinGraph}^l_\star$ by $h^\uparrow(v) = h_\downarrow(v)$, for $v \in I(T)$. In this case, we do not need to check that $h^\uparrow$ is well-defined, and the properties of mapping edges to edges, preserving the distinguished point, and monotonicity on neighborhoods, are inherited immediately from $h_\downarrow$. The fact that $h^\uparrow$ maps least paths to least paths is easily justified by observing that $h^\uparrow$ maps into $\Theta(G)$, the tree of least paths in G.

To check that this correspondence is bijective, notice that starting from either $h^\uparrow$ or $h_\downarrow$, passing to the other one, and passing back again, gives us the same morphism we started with. Naturality follows straightforwardly by observing that both functors are the identity on morphisms, so naturality squares trivially commute. ◀

▶ **Lemma 62.** *There is an adjunction* $\mathbf{LexGraph} \underset{I'}{\overset{\Theta'}{\top}} \mathbf{FinArb}^<_\star$.

Proof. The adjunction from Theorem 43 factors as

$$\mathbf{LexGraph} \overset{J}{\rightarrowtail} \mathbf{FinGraph}^l_\star \qquad I \dashv \Theta \qquad I' : \mathbf{FinGraph}^l_\star \to \mathbf{FinArb}^<_\star$$

where J is the fully faithful identity-on-objects functor witnessing the inclusion of **LexGraph** in $\mathbf{FinGraph}^l_\star$, and $\Theta' = \Theta \circ J$. The natural isomorphism

$$\begin{aligned}\mathbf{LexGraph}(I'(G), H) &\cong \mathbf{FinGraph}^l_\star(J(I'(G)), J(H)) \\ &= \mathbf{FinGraph}^l_\star(I(G), J(H)) \\ &\cong \mathbf{FinArb}^<_\star(G, \Theta(J(H))) \\ &= \mathbf{FinArb}^<_\star(G, \Theta'(H))\end{aligned}$$

establishes this adjunction. ◀

▶ **Lemma 63.** *For any finite, pointed, connected, edge-ordered graph G, $S(G)$ is an arborescence.*

Proof. To verify that $S(G)$ is a well-defined arborescence, it suffices to observe that $S(G)$ is connected (as it preserves least shortest paths), and that there is a unique path $v_0 \rightsquigarrow v$: two distinct edges $u \to v$ and $u' \to v$ cannot both occur in $\min^s(v_0 \rightsquigarrow v)$.

To check that $S(G \xrightarrow{h} H)$ is well defined, we have to verify for $u \to v \in S(G)$, $h(u) \to h(v) \in S(H)$. But such morphisms h preserve least shortest paths by Definition 34 (v). ◀

Proof of Theorem 45. The proof is obtained by literally copying the proof of Theorem 43 and replacing $\mathbf{FinGraph}^l_\star$ by $\mathbf{FinGraph}^s_\star$, Θ by S, and min by $\min^s$ throughout.

Given $I(T) \xrightarrow{h^\uparrow} G$ in $\mathbf{FinGraph}^s_\star$, we describe how to obtain $T \xrightarrow{h_\downarrow} T(G)$ in $\mathbf{FinArb}^<_\star$. Define $h_\downarrow(v) = h^\uparrow(v)$, for $v \in T$ (which we identified with $I(T)$). To check that $h_\downarrow$ is well-defined, we must show that if $u \to v$ is an edge of T, then $h^\uparrow(u \to v)$ is contained in $S(G)$. But since T is an arborescence, $u \to v$ is trivially contained in $\min^s(v_0 \rightsquigarrow v)$, so $h^\uparrow(u \to v)$ is contained in $\min^s(h(v_0) \rightsquigarrow h(v))$, and hence included in $S(G)$.

Moreover, the fact that $h_\downarrow$ maps edges to edges, preserves the distinguished point, and is monotone on co-initial edges, is immediately inherited from $h^\uparrow$.

Next, given $T \xrightarrow{h^\downarrow} S(G)$ in $\mathbf{FinArb}^<_\star$, define $I(T) \xrightarrow{h^\uparrow} G$ in $\mathbf{FinGraph}^s_\star$ by $h^\uparrow(v) = h_\downarrow(v)$, for $v \in I(T)$ (which we identified with T). In this case, we do not need to check that $h^\uparrow$ is well-defined, and the properties of mapping edges to edges, preserving the distinguished point, and monotonicity on neighborhoods, are inherited immediately from $h_\downarrow$. The fact that $h^\uparrow$ maps least shortest paths to least shortest paths is easily justified by observing that $h^\uparrow$ maps into $S(G)$, the tree of least shortest paths in G.

To check that this correspondence is bijective, notice that starting from either $h^\uparrow$ or $h_\downarrow$, passing to the other one, and passing back again, gives us the same morphism we started with. Naturality follows straightforwardly by observing that both functors are the identity on morphisms, so naturality squares trivially commute. ◀

Proof of Lemma 47. (i): If there are edges $u \to v_1$ and $u \to v_2$ of G, then $u \to v_1 \triangleleft u \to v_2$ in G iff $\min(u \rightsquigarrow v_1) \prec \min(u \rightsquigarrow v_2)$ in G (since G is a lex-graph) iff $u \to v_1 \triangleleft u \to v_2$ in $F(G)$ (by definition of F).

(ii): Suppose σ and π are co-initial paths from G. We may assume that σ and π share no nontrivial prefix; then either σ or π is empty (and the claim is trivial), or σ and π differ on their first edge, and the claim follows from (i).

(iii): Work in $F(G)$. It suffices to show that least paths in $F(G)$ consist of only edges in G; by the above remark, if two paths consist of G-edges, it does not matter whether we compare them in G or in $F(G)$.

Towards which, it suffices to show that every edge $u \to v$ not in G is greater than the least path $\min_G(u \rightsquigarrow v)$ between u and v in G. Then any path with non-edges in G can be lessened; in particular least paths in $F(G)$ cannot contain any non-edges of G.

Let $v_1 \neq v$ be the second vertex in $\min_G(u \rightsquigarrow v)$. Since least paths are closed under taking prefixes, $u \to v_1 = \min_G(u \rightsquigarrow v_1)$, in both G and $F(G)$. Since $\min_G(u \rightsquigarrow v_1) \prec \min_G(u \rightsquigarrow v)$, $u \to v_1 \triangleleft u \to v$ in $F(G)$. But then $\min_G(u \rightsquigarrow v) \prec u \to v$, which is what we wanted to show. ◀

Proof of Lemma 48. Given any lex-graph G, $F(G)$ is clearly a transitive edge-ordered graph, but we must check that it satisfies the lex-graph property (Definition 36-(ii)). But for any v_1 and v_2 in the neighborhood of a common u in $F(G)$, $u \to v_1 \triangleleft u \to v_2$ in $F(G)$ iff $\min(u \rightsquigarrow v_1) \prec \min(u \rightsquigarrow v_2)$ in G (by definition of F) iff $\min(u \rightsquigarrow v_1) \prec \min(u \rightsquigarrow v_2)$ in $F(G)$ (by Lemma 47 plus the preceding remark).

By definition, F immediately preserves identities and compositions of homomorphisms, and it remains to check that for any lex-homomorphism $G \xrightarrow{h} H$ of lex-graphs, $F(h)$ is a lex-homomorphism $F(G) \to F(H)$. We check each of the conditions in Definition 34:

- **(i)** If $u \to v$ in $F(G)$, then there is a path $u \rightsquigarrow v$ in G, so there is a path $h(u) \rightsquigarrow h(v)$ in H, and hence an edge $h(u) \to h(v)$ in $F(H)$.
- **(ii)** Since h preserves the distinguished point, so does $F(h)$.
- **(iii)** If $u \to v_1 \triangleleft u \to v_2$ in G, $\min(u \rightsquigarrow v_1) \prec \min(u \rightsquigarrow v_2)$, as G is a lex-graph. Since h is a lex-homomorphism, $\min_H(h(u) \rightsquigarrow h(v_i)) = h(\min_G(u \rightsquigarrow v_i))$, so $\min(h(u) \rightsquigarrow h(v_1)) \prec \min(h(u) \rightsquigarrow h(v_2))$ in H. By definition of F, $h(u) \to h(v_1) \triangleleft h(u) \to h(v_2)$ in $F(H)$.

(iv) As the least path $\min(u \rightsquigarrow v)$ in $F(G)$ is also least in G, as shown above, and as h preserves least paths, $h(\min(u \rightsquigarrow v))$ is $\min(h(u) \rightsquigarrow h(v))$ in H, and this in turn is also the least path in $F(H)$. ◀

Proof of Theorem 49. Let $F(G) \xrightarrow{h^\uparrow} H$ be a homomorphism of **TLexGraph**. Since $F(G)$ has the same vertices as G, we may define $G \xrightarrow{h_\downarrow} U(H)$ by $h_\downarrow(v) = h^\uparrow(v)$ (since H and $U(H)$ are exactly identical).

We check that $h_\downarrow$ is a lex-homomorphism by checking Definition 34 (i)-(iv). We write, e.g., $h(v)$ to refer unambiguously to the vertex $h^\uparrow(v) = h_\downarrow(v)$.

(i) If $u \to v$ is an edge of G, it is an edge of $F(G)$, so $h(u) \to h(v)$ is an edge of H, and hence an edge of $U(H)$.

(ii) $h_\downarrow$ directly inherits preservation of the distinguished point from $h^\uparrow$

(iii) If $u \to v_1 \lhd u \to v_2$ in G, then $u \to v_1 \lhd u \to v_2$ in $F(G)$ (Lemma 47), so $h(u) \to h(v_1) \lhd h(u) \to h(v_2)$ in H (monotonicity of $h^\uparrow$), and hence the same holds in $U(H)$.

(iv) If π is the least path $u \rightsquigarrow v$ in G, then it's least in $F(G)$ (Lemma 47), so $h(\pi)$ is least in H (since $h^\uparrow$ is a lex-homomorphism), and hence also in $U(H)$.

In the other direction, we suppose that we are given some homomorphism of lex-graphs $G \xrightarrow{h_\downarrow} U(H)$ in **LexGraph**. Define $F(G) \xrightarrow{h^\uparrow} H$ by $h^\uparrow(v) = h_\downarrow(v)$. Again, we check (i)-(iv) of Definition 34.

(i) If $u \to v$ is an edge of $F(G)$, then there is a path $u \rightsquigarrow v$ in G, and hence a path $h(u) \rightsquigarrow h(v)$ in $U(H)$, and (since H is transitive), and edge $h(u) \to h(v)$.

(ii) $h^\uparrow$ directly inherits preservation of the distinguished point from $h_\downarrow$

(iii) If $u \to v_1 \lhd u \to v_2$ in $F(G)$, then $\min(u \rightsquigarrow v_1) \prec \min(u \rightsquigarrow v_2)$ in G; hence (since $h_\downarrow$ is a lex-homomorphism) $\min(h(u) \rightsquigarrow h(v_1)) \prec \min(h(u) \rightsquigarrow h(v_2))$ in $U(H)$, hence in H. Since H is a lex-graph, $h(u) \to h(v_1) \lhd h(u) \to h(v_2)$.

(iv) If π is the least path $u \rightsquigarrow v$ in $F(G)$, then it's the least path in G (Lemma 47), so $h(\pi)$ is least in $U(H)$ (since $h_\downarrow$ is a lex-homomorphism), and hence in H.

We need only now to check that this correspondence is bijective and natural. Bijectivity follows readily from the fact that we always have $h^\uparrow = h_\downarrow$, so going from $G \xrightarrow{h_\downarrow} U(H)$ to $F(G) \xrightarrow{h^\uparrow} H$ and back has no effect, and similarly in the other direction. Similarly, naturality follows straightforwardly by noting that $U(h)(v) = F(h)(v) = h(v)$ for all h and v, so naturality squares trivially commute. ◀

Proof of Lemma 52. Notice that distances between vertices are never increased in G compared to T, only decreased, meaning that if there is a path $u \rightsquigarrow v$ in G, there is a path $u \rightsquigarrow v$ in T that is no shorter. Therefore, longest paths in G must consist entirely of edges in T, and are therefore unique.

Conversely, if $u \to v$ is an edge of T, then it appears on the unique path $v_0 \rightsquigarrow v$ in T. As just observed, the longest path $v_0 \rightsquigarrow v$ in G is also a path in T; hence, it is the unique path $v_0 \rightsquigarrow v$ in T, and thus contains $u \to v$. ◀

▶ **Lemma 64.** Γ *is a well-defined functor.*

Proof. Clearly Γ preserves identities and composition. We must check that for any morphism h of $\mathbf{FinArb}^<_\star$, $\Gamma(h)$ satisfies Definition 34 (i)-(iii) and preserves longest paths:

- (i) If $u \to v$ in $\Gamma(T)$, then $u \rightsquigarrow v$ in T, so $h(u) \rightsquigarrow h(v)$ in T', so $h(u) \to h(v)$ in $\Gamma(T')$.
- (ii) Since h maps the distinguished point, and only that point, of T to the distinguished point of T', Γ does the same from $\Gamma(T)$ to $\Gamma(T')$.

- (iii) If $u \to v_1 \triangleleft u \to v_2$ in $\Gamma(T)$, then $v_0 \rightsquigarrow v_1 \prec^s v_0 \rightsquigarrow v_2$ in T. By Lemma 35, $h(v_0 \rightsquigarrow v_1) \prec^s h(v_0 \rightsquigarrow v_2)$ in T' , and since paths in arborescences are unique, $h(u) \rightsquigarrow h(v_1) \prec^s h(u) \rightsquigarrow h(v_2)$ in T'. Hence $h(u) \to h(v_1) \triangleleft h(v) \to h(v_2)$ in $\Gamma(T')$.
- Finally, if $u \rightsquigarrow v$ is the longest path in $\Gamma(T)$, then it is a path in T by Lemma 52, and therefore, $h(u \rightsquigarrow v) = h(u) \rightsquigarrow h(v)$ is a path in T'. Again by Lemma 52, $h(u \rightsquigarrow v)$ is the longest path in $\Gamma(T')$. ◀

▶ **Lemma 65.** L *is a well-defined functor* $\mathbf{TArb} \to \mathbf{FinArb}^{<}_{\star}$.

Proof. In each graph $G \in \mathbf{TArb}$, the unique longest paths are closed under taking prefixes. Therefore, the union of all least longest paths forms an arborescence.

To check that L is a functor, note that for any morphism $h \in \mathbf{TArb}$, $\mathrm{L}(h)$ clearly preserves the distinguished point and is monotone on neighborhoods. We only need to show that if $u \to v$ is an edge in $\mathrm{L}(G)$ and $h : G \to H$ is a morphism in $\mathbf{TArb}$, then $h(u) \to h(v)$ is an edge of $\mathrm{L}(H)$. But, this guaranteed by the fact that h preserves longest paths.

Finally, note that L preserves the identity morphism and respects composition. ◀

Proof of Theorem 56. Fix a pointed, connected, edge-ordered arborescence T and a pointed, connected, transitive, edge-ordered graph G.

Given $\Gamma(T) \xrightarrow{h^\uparrow} G$ in $\mathbf{TArb}$, we define $T \xrightarrow{h_\downarrow} \mathrm{L}(G)$ by $h_\downarrow(v) = h^\uparrow(v)$. This is well-defined, because if $u \to v$ is an edge in T, then by Lemma 52, it appears on the unique longest path $v_0 \rightsquigarrow v$ in $\Gamma(T)$. Since $h^\uparrow$ preserves least longest paths, the edge $u \to v$ maps into $\mathrm{L}(G)$. Moreover, $h_\downarrow$ preserves the distinguished point and inherits monotonicity in neighborhoods from $h^\uparrow$, so satisfies the conditions of Definition 34 and is a morphism in $\mathbf{FinArb}^{<}_{\star}$.

In the other direction, given $T \xrightarrow{h_\downarrow} \mathrm{L}(G)$ in $\mathbf{FinArb}^{<}_{\star}$, we define $\Gamma(T) \xrightarrow{h^\uparrow} G$ by $h^\uparrow(v) = h_\downarrow(v)$; let us unambiguously write $h(v)$ for brevity. If $u \to v$ is an edge of $\Gamma(T)$, then there is a path $u \rightsquigarrow v$ in T, hence a path $h(u) \rightsquigarrow h(v)$ in $\mathrm{L}(G)$, and therefore an edge $h(u) \to h(v)$ in G. Moreover, if $u \to v_1 \triangleleft u \to v_2$ in $\Gamma(T)$, then $v_0 \rightsquigarrow v_1 \prec^s v_0 \rightsquigarrow v_2$ in T, so $h(u) \rightsquigarrow h(v_1) \prec^s h(u) \rightsquigarrow h(v_2)$ in $\mathrm{L}(G)$, by Lemma 35. By the remark succeeding Lemma 52, $h(u) \to h(v_1) \triangleleft h(u) \to h(v_2)$ in G. Finally, if $u \rightsquigarrow v$ is the longest path from u to v in $\Gamma(T)$, then each of its edges lies in T by Lemma 52, hence its h-image lies in $\mathrm{L}(G)$, which means it is a longest path of G. Therefore, $h^\uparrow$ is a morphism of $\mathbf{TArb}$.

It remains to show that the maps relating $h^\uparrow$ and $h_\downarrow$ are bijective and natural. As in the proof of Theorem 49, this is immediate from the definition of each map; the only thing to show is that they were well-defined. ◀

Proof of Lemma 57. Fix $u, v \neq v_0$. By definition of F, $v_0 \to u \triangleleft v_0 \to v$ in T iff $\min(v_0 \rightsquigarrow u) \prec \min(v_0 \rightsquigarrow v)$ in $(I' \circ \Theta)(G)$. Since I' is an inclusion functor, this is equivalent to $\min(v_0 \rightsquigarrow u) \prec \min(v_0 \rightsquigarrow v)$ in $\Theta(G)$; indeed, the *unique* path $v_0 \rightsquigarrow u$ is less than the *unique* path $v_0 \rightsquigarrow v$ in $\Theta(G)$.

But the unique paths $v_0 \rightsquigarrow u$ and $v_0 \rightsquigarrow v$ in $\Theta(G)$ are exactly $\min(v_0 \rightsquigarrow u)$ and $\min(v_0 \rightsquigarrow v)$ in G respectively; moreover, the relative order on the latter two paths in G is inherited from the relative order on the former two in $\Theta(G)$.

Therefore, $v_0 \to u \triangleleft v_0 \to v$ in T iff $\min(v_0 \rightsquigarrow u) \prec \min(v_0 \rightsquigarrow v)$ in G, but this is exactly the relation $<_D$ of Definition 28, which is the lexicographic depth-first traversal of G by Theorem 30. ◀

Proof of 58. Fix $u, v \neq v_0$. By definition of Γ, $v_0 \to u \triangleleft v_0 \to v$ in T iff $v_0 \rightsquigarrow u \prec^s v_0 \rightsquigarrow v$ in $S(G)$ (where paths from v_0 are unique). By definition of S, this is equivalent to $\min^s(v_0 \rightsquigarrow u) \prec^s \min^s(v_0 \rightsquigarrow v)$ in G. By Corollary 27, this is equivalent to $u <_B v$. ◀

Proof of Lemma 59. For a finite, edge-ordered graph G, we define $E(G)$ to be the order $(\{v_0\} \cup N(v_0), <)$, where for $u, v \neq v_0$, $u < v \iff v_0 \to u \triangleleft v_0 \to v$, and for $u \neq v_0$, $v_0 < u$.

On morphisms, given a homomorphism of edge-ordered graphs $G \xrightarrow{h} H$, we define $E(G) \xrightarrow{E(h)} E(H)$ by $E(h)(v) = h(v)$. Notice that E is well-defined, since it maps the distinguished point v_0 of G to the distinguished point w_0 of H, and also maps $N_G(v_0)$ into $N_H(w_0)$. By definition, it is clear that E preserves both identities and compositions, so we have only left to show that $E(h)$ is monotone.

Since h maps only v_0 to w_0, it suffices to show that if $u, v \in N_G(v_0)$ and $v_0 \to u \triangleleft v_0 \to v$ in G, then $w_0 \to h(u) \triangleleft w_0 \to h(v)$ in H. But this follows from monotonicity of h (Definition 34-(iii)) ◀