L-RECURSION AND A NEW LOGIC
FOR LOGARITHMIC SPACE [*]

MARTIN GROHE [a], BERIT GRUSSIEN [b], ANDRÉ HERNICH [c], AND BASTIAN LAUBNER [d]

[a] RWTH Aachen University, Germany
   e-mail address: grohe@informatik.rwth-aachen.de

[b,c,d] Humboldt-Universität, Berlin, Germany
   e-mail address: {grussien,hernich,laubner}@informatik.hu-berlin.de

ABSTRACT. We extend first-order logic with counting by a new operator that allows it to formalise a limited form of recursion which can be evaluated in logarithmic space. The resulting logic LREC has a data complexity in LOGSPACE, and it defines LOGSPACE-complete problems like deterministic reachability and Boolean formula evaluation. We prove that LREC is strictly more expressive than deterministic transitive closure logic with counting and incomparable in expressive power with symmetric transitive closure logic STC and transitive closure logic (with or without counting). LREC is strictly contained in fixed-point logic with counting FP+C. We also study an extension LREC$_=$ of LREC that has nicer closure properties and is more expressive than both LREC and STC, but is still contained in FP+C and has a data complexity in LOGSPACE.

Our main results are that LREC captures LOGSPACE on the class of directed trees and that LREC$_=$ captures LOGSPACE on the class of interval graphs.

1. INTRODUCTION

Descriptive complexity theory gives logical characterisations for most of the standard complexity classes. For example, Fagin's Theorem [7] states that a property of finite structures is decidable in NP if and only if it is definable in existential second-order logic Σ_1^1. More concisely, we say that Σ_1^1 *captures* NP. Similarly, Immerman [13] and Vardi [26] proved that fixed-point logic FP captures PTIME,[1] and Immerman [15] proved that deterministic transitive closure logic DTC captures LOGSPACE. However, these and all other known logical

[1]More precisely, Immerman and Vardi's theorem holds for *least fixed-point logic* and the equally expressive *inflationary fixed-point logic.* Our indeterminate FP refers to either of the two logics. For the counting extension FP+C considered below, it is most convenient to use an inflationary fixed-point operator. See any of the textbooks [5, 10, 16, 22] for details.

characterisations of PTIME and LOGSPACE and all other complexity classes below NP have a serious drawback — they only hold on ordered structures. (An *ordered structure* is a structure that has a distinguished binary relation which is a linear order of the elements of the structure.) The question of whether there are logical characterisations of these complexity classes on arbitrary, not necessarily ordered structures, is viewed as the most important open problem in descriptive complexity theory. For the class PTIME this problem goes back to Chandra and Harel's fundamental article [4] on query languages for relational databases.

For PTIME, at least partial positive results are known. The strongest of these say that fixed-point logic with counting FP+C captures PTIME on all classes of graphs with excluded minors [11] and on the class of interval graphs [19]. It is well-known that fixed-point logic FP (without counting) is too weak to capture PTIME on any natural class of structures that are not ordered. The idea that the extension FP+C by counting operators might remedy the weakness of FP goes back to Immerman [14]. Together with Lander he proved that FP+C captures PTIME on the class of trees [17]. Later, Cai, Fürer, and Immerman [3] proved that FP+C does not capture PTIME on all finite structures.

Much less is known for LOGSPACE. In view of the results described so far, an obvious idea is to try to capture LOGSPACE with the extension DTC+C of deterministic transitive closure logic DTC by counting operators. However, Etessami and Immerman [6] proved that (directed) tree isomorphism is not definable in DTC+C, not even in the stronger transitive closure logic with counting TC+C. Since Lindell [23] proved that tree isomorphism is decidable in LOGSPACE, this shows that DTC+C does not capture LOGSPACE.

We introduce a new logic LREC and prove that it captures LOGSPACE on directed trees. An extension LREC$_=$ captures LOGSPACE on the class of interval graphs (and on the class of undirected trees). The logic LREC is an extension of first-order logic with counting by a "limited recursion operator". The logic is more complicated than the transitive closure and fixed-point logics commonly studied in descriptive complexity, and it may look rather artificial at first sight. To explain the motivation for this logic, recall that fixed-point logics may be viewed as extensions of first-order logic by fixed-point operators that allow it to formalise recursive definitions in the logics. LREC is based on an analysis of the amount of recursion allowed in logarithmic space computations. The idea of the limited recursion operator is to control the depth of the recursion by a "resource term", thereby making sure that we can evaluate the recursive definition in logarithmic space. Another way to arrive at the logic is based on an analysis of the classes of Boolean circuits that can be evaluated in LOGSPACE. We will take this route when we introduce the logic in Section 3.

LREC is easily seen to be (semantically) contained in FP+C. We show that LREC contains DTC+C, and as LREC captures LOGSPACE on directed trees, this containment is strict. Moreover, LREC is not contained in TC+C. Then we prove that undirected graph reachability is not definable in LREC. Hence LREC does not contain transitive closure logic TC, not even in its symmetric variant STC, and therefore LREC is strictly contained in FP+C.

It can be argued that our proof of the inability of LREC to express graph reachability reveals a weakness in our definition of the logic rather than a weakness of the limited recursion operator underlying the logic: LREC is not closed under (first-order) logical reductions. To remedy this weakness, we introduce an extension LREC$_=$ of LREC. It turns out that undirected graph reachability is definable in LREC$_=$ (this is a convenient side effect of the definition and not a deep result). Thus LREC$_=$ strictly contains symmetric transitive closure logic with counting. We prove that LREC$_=$ captures LOGSPACE on the class of interval graphs. To complete the picture, we prove that plain LREC, even if extended by a

symmetric transitive closure operator, does not capture LOGSPACE on the class of interval graphs.

The paper is organised as follows: After giving the necessary preliminaries in Section 2, in Section 3 we introduce the logic LREC and prove that its data complexity is in LOGSPACE. Then in Section 4, we prove that directed tree isomorphism and canonisation are definable in LREC. As a consequence, LREC captures LOGSPACE on directed trees. In Section 5, we study the expressive power of LREC and prove that undirected graph reachability is not definable in LREC. The extension LREC$_=$ is introduced in Section 6. Finally, our results on interval graphs are presented in Section 7. We close with a few concluding remarks and open problems.

2. Basic Definitions

$\mathbb{N}$ denotes the set of all non-negative integers. For all $m, n \in \mathbb{N}$, we let $[m, n] := \{p \in \mathbb{N} \mid m \le p \le n\}$ and $[n] := [1, n]$. Mappings $f\colon A \to B$ are extended to tuples $\bar{a} = (a_1, \ldots, a_k)$ over A via $f(\bar{a}) := (f(a_1), \ldots, f(a_k))$. Given a tuple $\bar{a} = (a_1, \ldots, a_k)$, let $\tilde{a} := \{a_1, \ldots, a_k\}$. If $\sim$ is an equivalence relation on a set A, we denote by $a/_\sim$ the equivalence class of an element a with respect to $\sim$, and by $A/_\sim$ the quotient of A with respect to $\sim$.

A *vocabulary* is a finite set τ of relation symbols, where each $R \in \tau$ has a fixed arity $\mathrm{ar}(R)$. A τ-*structure* A consists of a non-empty finite set $V(A)$, its *universe*, and for each $R \in \tau$ a relation $R(A) \subseteq V(A)^{\mathrm{ar}(R)}$. For logics L, L' we write $\mathsf{L} \le \mathsf{L}'$ if L is semantically contained in L', and $\mathsf{L} < \mathsf{L}'$ if this containment is strict.

All logics considered in this paper are extensions of *first-order logic with counting* (FO+C); see, e.g., [5, 10, 16, 22, 15] for a detailed discussion of FO+C and its extensions. FO+C extends first-order logic by a counting operator that allows for counting the cardinality of FO+C-definable relations. It lives in a two-sorted context, where structures A are equipped with a *number sort* $N(A) := [0, |V(A)|]$. FO+C-variables are either *structure variables* that range over the universe of a structure, or *number variables* that range over the number sort. For each variable u, let $A^u := V(A)$ if u is a structure variable, and $A^u := N(A)$ if u is a number variable. Tuples $(u_1, \ldots, u_k)$ and $(v_1, \ldots, v_\ell)$ of variables are *compatible* if $k = \ell$, and for every $i \in [k]$ the variables u_i and v_i have the same type. Let $A^{(u_1, \ldots, u_k)} := A^{u_1} \times \cdots \times A^{u_k}$. An *assignment in A* is a mapping α from the set of variables to $V(A) \cup N(A)$, where for each variable u we have $\alpha(u) \in A^u$. For tuples $\bar{u} = (u_1, \ldots, u_k)$ of variables and $\bar{a} = (a_1, \ldots, a_k) \in A^{\bar{u}}$, the assignment $\alpha[\bar{a}/\bar{u}]$ maps u_i to a_i for each $i \in [k]$, and each variable $v \notin \tilde{u}$ to $\alpha(v)$.

FO+C is obtained by extending first-order logic with the following formula formation rules: $p \le q$ is a formula for all number variables p, q; and $\#\bar{u}\,\psi = \bar{p}$ is a formula for all tuples $\bar{u}$ of variables, all tuples $\bar{p}$ of number variables, and all formulae ψ. Free variables are defined in the obvious way, with $\mathrm{free}(\#\bar{u}\,\psi = \bar{p}) := (\mathrm{free}(\psi) \setminus \tilde{u}) \cup \tilde{p}$. Formulae $\#\bar{u}\,\psi = \bar{p}$ hold in a structure A under an assignment α in A if $|\{\bar{a} \in A^{\bar{u}} \mid (A, \alpha[\bar{a}/\bar{u}]) \models \psi\}| = \langle\alpha(\bar{p})\rangle_A$, where for tuples $\bar{n} = (n_1, \ldots, n_k) \in N(A)^k$ we let $\langle\bar{n}\rangle_A$ be the number

$$\langle\bar{n}\rangle_A \; := \; \sum_{i=1}^{k} n_i \cdot (|V(A)| + 1)^{i-1}.$$

If A is understood from the context, we write $\langle\bar{n}\rangle$ instead of $\langle\bar{n}\rangle_A$.

We write $\varphi(u_1, \ldots, u_k)$ to denote a formula φ with $\mathrm{free}(\varphi) \subseteq \{u_1, \ldots, u_k\}$. Given a formula $\varphi(u_1, \ldots, u_k)$, a structure A and $a_1, \ldots, a_k \in A^{(u_1, \ldots, u_k)}$, we write $A \models \varphi[a_1, \ldots, a_k]$ if φ holds in A with u_i assigned to the element a_i, for each $i \in [k]$. We use similar notation for substitution: For a tuple $(v_1, \ldots, v_k)$ of variables that is compatible with $(u_1, \ldots, u_k)$, we let $\varphi(v_1, \ldots, v_k)$ be the result of substituting v_i for u_i for every $i \in [k]$. We write $\varphi[A, \alpha; \bar{u}]$ for the set of all tuples $\bar{a} \in A^{\bar{u}}$ with $(A, \alpha[\bar{a}/\bar{u}]) \models \varphi$.
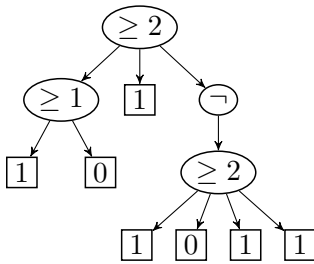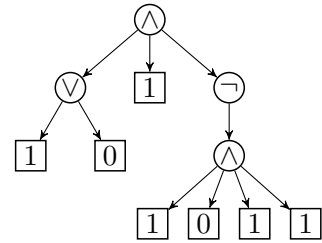
In many places throughout this paper we refer to various transitive closure and fixed-point logics (all mentioned in the introduction). Our results and remarks about the relation between these logics and our new logics LREC and LREC$_=$ are relevant for a reader familiar with descriptive complexity theory to put our results in context, but they are not essential to follow the technical core of this paper. Therefore, we omit the definitions and refer the reader to the textbooks [5, 10, 16, 22] and the paper [15].

3. The Logic LREC

In this section, we introduce LREC as a first step towards the logic LREC$_=$, to be introduced in Section 6. LREC is already expressive enough to capture LOGSPACE on directed trees, but still lacks several important properties. For example, it is unable to capture LOGSPACE on undirected trees and interval graphs (cf. Remark 7.15), and is not closed under first-order reductions (Section 6). On the other hand, although LREC$_=$ could have been introduced without the detour via LREC, its definition is much easier to grasp by developing an understanding of LREC first.

Let us start our development of LREC by looking at how certain kinds of Boolean circuits can be evaluated in LOGSPACE.

The figure on the right shows a *Boolean formula*, i.e., a Boolean circuit whose underlying graph is a *tree*. It is easy to evaluate such circuits in LOGSPACE: Start at the output node, determine the value of the first child recursively, then determine the value of the second child, and so on. We only have to store the current node and its value (if it has been determined already), since the parent node and the next child of the parent (if any) are uniquely determined by the current node. It is known that Boolean formula evaluation is complete for LOGSPACE under NC[1]-reductions [1].[2] In contrast, Boolean *circuit* evaluation is PTIME-complete.
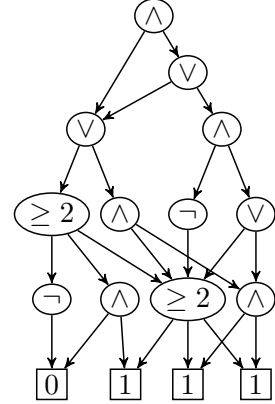

Let us now turn to formulae with *threshold gates*, which, in addition to Boolean gates, may contain gates of the form "$\geq i$" for a number i; such a gate outputs 1 if, and only if, at least i input gates are set to 1. An example is shown on the left. To evaluate such formulae in LOGSPACE, we again start at the root and evaluate the values of the children recursively. For each node we count how many 1-values we have seen already. To this end, when evaluating the values of the children of a node v, we begin with the child with the largest subtree and proceed to children with smaller subtrees. Note that the ith

---

[2]Boolean formula evaluation is only complete for LOGSPACE if input formulae are represented as graphs (e.g., by the list of all edges plus gate types). It was however shown in [2] that the problem is complete for NC[1] under AC[0]-reductions if input formulae are given by their natural string encoding.

child of v in this order has a subtree of size at most s/i, where s is the size of the subtree of v. So, we can store a counter of up to $\log_2 i$ bits for the number of 1-values seen so far. It is easy to extend the algorithm to formulae with other *arithmetic gates* such as *modulo-gates*.

As a more complicated example, let us consider the following type of circuit. A circuit C has the *m-path property* if for all paths P in C the product of the in-degrees of all but the first node on P is at most m. For example, formulae have the 1-path property, whereas the circuit on the right has the 16-path property. It is not hard to see that for every $k \geq 1$, circuits C having the $|C|^k$-path property can be evaluated in LOGSPACE. The idea here is very similar to the one for evaluating circuits with threshold gates. We start at the root node and evaluate the children recursively. After "entering" a node v from one of its parent nodes, say $p(v)$, we check whether v evaluates to 1 by counting the number of children that evaluate to one using the above-mentioned strategy, and return with this information to $p(v)$. In order to return to $p(v)$, we need to remember $p(v)$, which we do by storing the index of $p(v)$ among all the in-neighbours of v. This requires only $\log_2 d^-(v)$ bits of storage, where $d^-(v)$ denotes the in-degree of v. The space for writing down the index of the predecessor $p(v)$ for each vertex v on the path from the root to the currently visited vertex is thus bounded by the sum of the logarithms of the in-degrees of the vertices v on that path. Since C has the $|C|^k$-path property, this sum is bounded by $\log_2|C|^k$, and thus logarithmic in the size of C. Another way of evaluating the circuit is to first "unravel" the circuit to a tree (i.e., a formula) which can be done in LOGSPACE due to the $|C|^k$-path property, and then to evaluate the formula as above.

The logic LREC allows it to recursively define sets X of tuples based on graphs G that have the $|G|^k$-path property for some $k \geq 1$.

We turn to the formal definition of the logic LREC. To define the syntax, let τ be a vocabulary. The set of all LREC$[\tau]$-formulae is obtained by extending the formula formation rules of FO+C$[\tau]$ by the following rule: If $\bar{u}, \bar{v}, \bar{w}$ are compatible tuples of variables, $\bar{p}, \bar{r}$ are non-empty tuples of number variables, and φ_E and φ_C are LREC$[\tau]$-formulae, then

$$\varphi := [\text{lrec}_{\bar{u}, \bar{v}, \bar{p}} \, \varphi_E, \, \varphi_C](\bar{w}, \bar{r}) \tag{3.1}$$

is an LREC$[\tau]$-formula, and we let $\text{free}(\varphi) := (\text{free}(\varphi_E) \setminus (\tilde{u} \cup \tilde{v})) \cup (\text{free}(\varphi_C) \setminus (\tilde{u} \cup \tilde{p})) \cup \tilde{w} \cup \tilde{r}$.

To define the semantics of LREC$[\tau]$-formulae, let A be a τ-structure and α an assignment in A. The semantics of LREC$[\tau]$-formulae that are not of the form (3.1) is defined as usual.

Let φ be an LREC$[\tau]$-formula of the form (3.1). We define a set $X \subseteq A^{\bar{u}} \times \mathbb{N}$ recursively as follows. We consider $E := \varphi_E[A, \alpha; \bar{u}, \bar{v}]$ as the edge relation of a directed graph G with vertex set $V := A^{\bar{u}}$. Moreover, for each vertex $\bar{a} \in V$ we think of the set $C(\bar{a}) := \{\langle \bar{n} \rangle \mid \bar{n} \in \varphi_C[A, \alpha[\bar{a}/\bar{u}]; \bar{p}]\}$ of integers as the label of $\bar{a}$. Let $\bar{a}E := \{\bar{b} \in V \mid \bar{a}\bar{b} \in E\}$ and $E\bar{b} := \{\bar{a} \in V \mid \bar{a}\bar{b} \in E\}$. Then, for all $\bar{a} \in V$ and $\ell \in \mathbb{N}$,

$$(\bar{a}, \ell) \in X \; :\Longleftrightarrow \; \ell > 0 \text{ and } \left| \left\{ \bar{b} \in \bar{a}E \; \middle| \; \left(\bar{b}, \left\lfloor \frac{\ell - 1}{|E\bar{b}|} \right\rfloor \right) \in X \right\} \right| \in C(\bar{a}).$$

Notice that X contains only elements $(\bar{a}, \ell)$ with $\ell > 0$. Hence, the recursion eventually stops at $\ell = 0$. We call X the *relation defined by φ in (A, α)*. Finally, we let

$$(A, \alpha) \models \varphi \; :\Longleftrightarrow \; (\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle) \in X.$$
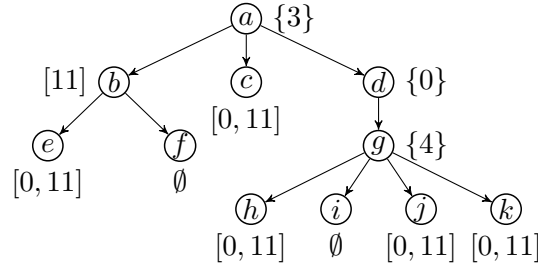
Figure 1: The graph G from Example 3.1. Each vertex is labelled with a subset of $[0, 11]$.

Example 3.1 (Boolean circuit evaluation). Let $\sigma := \{E, P_\wedge, P_\vee, P_\neg, P_0, P_1\}$. A Boolean circuit C may be viewed as a σ-structure, where $E(C)$ is the edge relation of C, and $P_\star(C)$ contains all $\star$-gates for $\star \in \{\wedge, \vee, \neg, 0, 1\}$. Suppose C has the $|C|$-path-property. Then,

$$\varphi(z) := \exists r_1, r_2 \left([\mathsf{lrec}_{x,y,p} \; \varphi_{\mathtt{E}}, \; \varphi_{\mathtt{C}}](z, (r_1, r_2)) \wedge \forall r (r \leq r_1 \wedge r \leq r_2) \right)$$

with $\varphi_{\mathtt{E}}(x, y) := E(x, y)$ and

$$\varphi_{\mathtt{C}}(x, p) := (P_\wedge(x) \wedge \#y \, E(x, y) = p) \vee (P_\vee(x) \wedge \text{``}p > 0\text{''}) \vee (P_\neg(x) \wedge \text{``}p = 0\text{''}) \vee P_1(x)$$

states that gate z evaluates to 1.

For example, let C be the first circuit at the beginning of this section, and let α be the assignment in C mapping z to the root of C, r_1 to 4, and r_2 to 0. Figure 1 shows the graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ with $\mathsf{V} := C^x$, $\mathsf{E} := \varphi_{\mathtt{E}}[C, \alpha; x, y]$, and labels defined by $\varphi_{\mathtt{C}}$. The vertices a–k of G are precisely the vertices of C, and each vertex is labelled with a subset of $N(C) = [0, 11]$. Let X be the relation defined by $[\mathsf{lrec}_{x,y,p} \; \varphi_{\mathtt{E}}, \; \varphi_{\mathtt{C}}](z, (r_1, r_2))$ in (C, α). For a leaf v of G, we have $(v, 1) \in X$ (and, in fact, $(v, \ell) \in X$ for any $\ell > 0$) if and only if 0 occurs in the label of v. Hence, $(v, 1) \in X$ for $v \in \{c, e, h, j, k\}$, but $(f, 1) \notin X$ and $(i, 1) \notin X$. Since $(e, 1) \in X$ and 1 occurs in the label of b, we also have $(b, 2) \in X$; as for the leaves, we also have $(b, \ell) \in X$ for any $\ell \geq 2$. However, note that $(g, 2) \notin X$ (and, in fact, $(g, \ell) \notin X$ for all $\ell > 0$), because there are only three children v of g with $(v, 1) \in X$, but 3 does not appear in the label of g. Consequently, $(d, 3) \in X$. Since we now have $(b, 3) \in X$, $(c, 3) \in X$, and $(d, 3) \in X$, we have $(a, 4) \in X$, and therefore $(C, \alpha) \models \varphi$.

While for the circuit C above, we could have replaced the tuple (r_1, r_2) in the formula φ by a single number variable r, it is not hard to construct circuits C which have the $|C|$-path property, but the single number variable r does not suffice. $\qquad \square$

Example 3.2 (Deterministic transitive closure). Let $G = (V, E)$ be a directed graph and $a, b \in V$. Then there is a *deterministic path* from a to b in G if there exists a path $v_1, \ldots, v_n$ from $a = v_1$ to $b = v_n$ in G such that for every $i \in [n-1]$, v_{i+1} is the unique out-neighbour of v_i. Figure 2(a) shows a directed graph with a deterministic path from c to d.

Let $\psi(\bar{u}, \bar{v})$ be an $\mathsf{LREC}[\tau]$-formula, and let $\bar{s}, \bar{t}$ be tuples of variables such that $\bar{u}, \bar{v}, \bar{s}, \bar{t}$ are pairwise compatible. We devise a formula $\varphi(\bar{s}, \bar{t})$ such that for any τ-structure A and assignment α in A, we have $(A, \alpha) \models \varphi(\bar{s}, \bar{t})$ iff in the graph $G = (V, E)$ defined by $V := A^{\bar{u}}$ and $E := \psi[A, \alpha; \bar{u}, \bar{v}]$ there is a deterministic path from $\alpha(\bar{s})$ to $\alpha(\bar{t})$. Note that there is such a path precisely if, in the graph obtained from G by reversing the edges, there is a path $v_n, \ldots, v_1$ from $\alpha(\bar{t})$ to $\alpha(\bar{s})$ such that for every $i \in [n-1]$, v_{i+1} is the unique in-neighbour
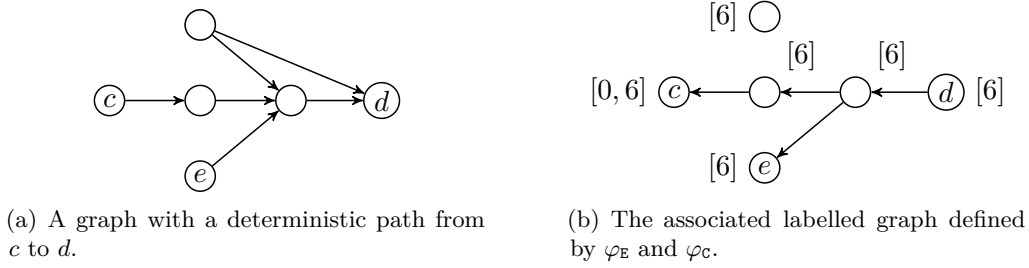
(a) A graph with a deterministic path from c to d.

(b) The associated labelled graph defined by $\varphi_{\mathtt{E}}$ and $\varphi_{\mathtt{C}}$.

Figure 2: A graph with a deterministic path, and the labelled graph defined by the formulae $\varphi_{\mathtt{E}}$ and $\varphi_{\mathtt{C}}$ in Example 3.2 from that graph.

of v_i. Therefore, we can choose φ like this:

$$\varphi \ := \ \exists \bar{r}\, [\mathsf{lrec}_{\bar{v},\bar{u},\bar{p}} \ \varphi_{\mathtt{E}}(\bar{v},\bar{u}), \ \varphi_{\mathtt{C}}(\bar{v},\bar{p})](\bar{t},\bar{r}), \tag{3.2}$$

where $\bar{p}$ and $\bar{r}$ are $|\bar{u}|$-tuples of number variables, and

$$\varphi_{\mathtt{E}}(\bar{v},\bar{u}) \ := \ \psi(\bar{u},\bar{v}) \wedge \forall \bar{v}'(\psi(\bar{u},\bar{v}') \rightarrow \bar{v}' = \bar{v}), \qquad \varphi_{\mathtt{C}}(\bar{v},\bar{p}) \ := \ \bar{v} = \bar{s} \vee (\bar{v} \neq \bar{s} \wedge \bar{p} \neq \bar{0}).$$

Informally, $\varphi_{\mathtt{E}}(\bar{v},\bar{u})$ removes all edges $\bar{a}\bar{b}$ of G, where $\bar{a}$ has more than one out-neighbour, and reverses the remaining edges. All that remains is to check whether there is a path from $\alpha(\bar{t})$ to $\alpha(\bar{s})$ in the graph defined by $\varphi_{\mathtt{E}}$. The node labelling formula $\varphi_{\mathtt{C}}$ is chosen in such a way that the latter is true iff $(\alpha(\bar{t}), \ell)$, for an $\ell \leq |V|$, appears in the relation X defined by φ in (A,α). If, for example, G is the graph in Figure 2(a), and if $\alpha(\bar{s}) = c$ and $\alpha(\bar{t}) = d$, then the labelled graph defined by $\varphi_{\mathtt{E}}$ and $\varphi_{\mathtt{C}}$ is as shown in Figure 2(b), and it is easy to see that $(d,4) \in X$, while, for example, $(e,\ell) \notin X$ for all $\ell > 0$. $\qquad\square$

As from now, we use

$$[\mathsf{dtc}_{\bar{u},\bar{v}} \ \psi](\bar{s},\bar{t}) \tag{3.3}$$

as an abbreviation for the LREC-formula in (3.2).

Remark 3.3. In the preceding two examples, the set X turned out to possess a certain monotonicity property: If $(\bar{a},\ell) \in X$ for some ℓ, then $(\bar{a},\ell') \in X$ for all $\ell' \geq \ell$. In general, however, the relation X defined by an lrec operator does not possess this property. For example, consider the formula $\varphi := [\mathsf{lrec}_{u,v,p} \ E(u,v), \ \text{"}p = 0\text{"}](u,p)$. Now let G be the graph consisting of a single edge (a,b), and let α be the assignment mapping u to a and p to 2. Then the relation X defined by φ in (G,α) contains $(a,1)$, but not $(a,2)$.

The following theorem shows that the data complexity of LREC is in LOGSPACE.

Theorem 3.4. *For every vocabulary τ, and every $\mathsf{LREC}[\tau]$-formula φ there is a deterministic logspace Turing machine that, given a τ-structure A and an assignment α in A, decides whether $(A,\alpha) \models \varphi$.*

Proof. We proceed by induction on the structure of φ. The case where φ is *not* of the form (3.1) is easy. Let φ be of the form (3.1), i.e., let

$$\varphi \ = \ [\mathsf{lrec}_{\bar{u},\bar{v},\bar{p}} \ \varphi_{\mathtt{E}}, \ \varphi_{\mathtt{C}}](\bar{w},\bar{r}).$$

Let $\mathtt{G} = (\mathtt{V},\mathtt{E})$ be the graph with $\mathtt{V} = A^{\bar{u}}$ and $\mathtt{E} = \varphi_{\mathtt{E}}[A,\alpha;\bar{u},\bar{v}]$, let $\mathtt{C}(\bar{a}) := \{\langle \bar{n}\rangle \mid \bar{n} \in \varphi_{\mathtt{C}}[A,\alpha[\bar{u}/\bar{a}];\bar{p}]\}$ for all $\bar{a} \in \mathtt{V}$, and let $X \subseteq \mathtt{V} \times \mathbb{N}$ be the relation defined by φ in (A,α). We construct a deterministic logspace Turing machine that decides whether $(\alpha(\bar{w}), \langle\alpha(\bar{r})\rangle) \in X$.

The machine is constructed in two steps. The first step consists of constructing a deterministic logspace Turing machine M_1 that, given A and α as input, computes a labelled directed tree T that is obtained basically from "unravelling" G starting at $\alpha(\bar{w})$ with "resource" $\langle \alpha(\bar{r}) \rangle$. The second step is to devise a deterministic logspace Turing machine M_2 that takes T as input and decides whether its root, $(\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle)$, belongs to X. The composition of M_1 and M_2 finally yields the desired machine.

Let $k := |\bar{r}|$. We define a labelled directed tree T whose set W of vertices consists of all the sequences $((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m))$ of pairs from $\mathsf{V} \times \mathbb{N}$ for some $m \in \mathbb{N}$ such that

(1) $(\bar{a}_0, \ell_0) = (\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle)$,
(2) $\bar{a}_{i+1} \in \bar{a}_i \mathsf{E}$ for all $i < m$, and
(3) $\ell_{i+1} = \left\lfloor \frac{\ell_i - 1}{|\mathsf{E}\bar{a}_{i+1}|} \right\rfloor$ for all $i < m$.

There is an edge from $((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m))$ to $((\bar{a}'_0, \ell'_0), \ldots, (\bar{a}'_{m'}, \ell'_{m'}))$ in T if $m' = m+1$, and $(\bar{a}'_i, \ell'_i) = (\bar{a}_i, \ell_i)$ for all $i \leq m$. We label each vertex $v = ((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m)) \in W$ with the set $\mathsf{C}(v) := \mathsf{C}(\bar{a}_m)$, and with the number $fail(v) \in \{0, 1\}$ such that $fail(v) = 1$ iff $\ell_m = 0$. Note that $fail(v) = 1$ only if v is a leaf in T. Clearly, T is a labelled directed tree rooted at $(\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle)$.

Define $Y \subseteq W$ such that

$$v \in Y \iff |\{w \in Y \mid w \text{ is a child of } v\}| \in \mathsf{C}(v) \ \text{ and } \ fail(v) = 0 \quad \text{(for every } v \in W).$$

Claim 1. For every $v = ((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m)) \in W$ we have $v \in Y$ if and only if $(\bar{a}_m, \ell_m) \in X$. In particular, $(\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle) \in X$ if and only if $(\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle) \in Y$.

Proof. The proof is by induction on the *rank* r_v of v in T: if v is a leaf in T, then $r_v = 0$; and if v is not a leaf in T, then r_v is one more than the maximum of the ranks of v's children. For every $v = ((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m)) \in W$, let $\lambda(v) := (\bar{a}_m, \ell_m)$.

Suppose that $r_v = 0$, that is, v is a leaf in T. Consider $(\bar{a}, \ell) = \lambda(v)$. Then $\bar{a}\mathsf{E}$ is the empty set or $\ell = 0$. First consider the case that $\ell = 0$. In this case, $(\bar{a}, \ell) \notin X$ by the definition of X. But we also have $fail(v) = 1$, which implies $v \notin Y$. Next consider the case that $\bar{a}\mathsf{E}$ is the empty set and $\ell > 0$. In this case,

$$v \in Y \iff 0 \in \mathsf{C}(v) = \mathsf{C}(\bar{a}) \iff (\bar{a}, \ell) \in X,$$

as desired.

Suppose now that $r_v = r + 1$, and that the claim is true for vertices w with $r_w \leq r$. In particular, since v is not a leaf we must have $fail(v) = 0$. This implies $\ell > 0$, and

$$v \in Y \iff |\{w \in Y \mid w \text{ is a child of } v\}| \in \mathsf{C}(v)$$
$$\iff |\{\lambda(w) \in X \mid w \text{ is a child of } v\}| \in \mathsf{C}(v) \quad \text{by the induction hypothesis.} \quad (3.4)$$

Let W' be the set of all children w of v such that $\lambda(w) \in X$, and let $f \colon W' \to A^{\bar{u}}$ be such that for all $w \in W'$, $f(w)$ is the first component of $\lambda(w)$. Then f is a bijection from W' to the set of all tuples $\bar{b} \in \bar{a}\mathsf{E}$ with

$$\left(\bar{b}, \left\lfloor \frac{\ell - 1}{|\mathsf{E}\bar{b}|} \right\rfloor \right) \in X. \tag{3.5}$$

As a consequence, the number of all tuples $\bar{b} \in \bar{a}\mathsf{E}$ with (3.5) is precisely $|W'|$. Hence, by (3.4) and $\ell > 0$,

$$v \in Y \iff |W'| \in \mathsf{C}(v) = \mathsf{C}(\bar{a}) \iff \lambda(v) = (\bar{a}, \ell) \in X. \qquad \square$$

By Claim 1, it suffices to compute T, and use T to decide whether its root, $(\alpha(\bar{w}), \langle\alpha(\bar{r})\rangle)$, belongs to Y. This is precisely what the two machines M_1 and M_2 mentioned at the beginning of this proof do. We now prove the existence of such machines.

Claim 2. There is a deterministic logspace Turing machine that takes A and α as input and outputs T.

Proof. We first construct a deterministic logspace Turing machine M that takes A and α as input and outputs the vertices of T (represented as sequences $((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m))$ as above). This machine makes use of a deterministic logspace Turing machine M_{E} that takes A, α and a pair $(\bar{a}, \bar{b}) \in \mathtt{V}^2$ as input and decides whether $\bar{a}\bar{b} \in \mathtt{E}$. Such a machine exists by the induction hypothesis. Once M is constructed, we can easily compute the edges and the labels of T, using a deterministic logspace Turing machine for computing the labels $\mathtt{C}(\bar{a})$ for each $\bar{a} \in \mathtt{V}$ as guaranteed by the induction hypothesis.

In what follows, we describe how M computes the vertices of T from A and α. We basically do a depth-first search in $\mathtt{G}$ starting in $\alpha(\bar{w})$ with "resources" $\langle\alpha(\bar{r})\rangle$. In each step, we visit some vertex $\bar{a} \in \mathtt{V}$. We also maintain a number $\ell < |N(A)|^k$, the length m of the path $P = (\bar{a}_0, \ldots, \bar{a}_m)$ on which $\bar{a}$ was reached from $\alpha(\bar{w})$, and for each $i \in [m]$ a number $e_i \in [0, |\mathtt{E}\bar{a}_i| - 1]$ with the following property. For each $\bar{b} \in A^{\bar{u}}$ let $\bar{b}_0, \ldots, \bar{b}_p$ be the elements of $\mathtt{E}\bar{b}$ ordered lexicographically according to their representation in the input string; let $\mathrm{pre}(\bar{b}, i) := \bar{b}_i$. Then the number e_i will have the property that $\bar{a}_{i-1} = \mathrm{pre}(\bar{a}_i, e_i)$. When we move from $\bar{a}$ to some vertex $\bar{b} \in \bar{a}\mathtt{E}$ we update ℓ to be

$$\mathrm{decr}(\ell, \bar{b}) := \left\lfloor \frac{\ell - 1}{|\mathtt{E}\bar{b}|} \right\rfloor .$$

This ensures that the space needed to store the numbers $e_1, \ldots, e_m$ is logarithmic in $|W|$ (which we shall prove later). Finally, upon visiting $\bar{a}$ for the first time, we write the sequence $(\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m)$ to the output tape, where the ℓ_i are the values for ℓ maintained along the path P.

More precisely, we proceed as follows. In the first step, we let $\bar{a} := \alpha(\bar{w})$, $\ell := \langle\alpha(\bar{r})\rangle$ and $m := 0$. Let $\bar{a} \in \mathtt{V}$, $\ell < |N(A)|^k$, $m \in \mathbb{N}$ and numbers $e_1, \ldots, e_m$ be given. Furthermore, let $\bar{a}_0, \ldots, \bar{a}_m$ be such that $\bar{a}_m = \bar{a}$, and for each $i \in [m]$, $\bar{a}_{i-1} = \mathrm{pre}(\bar{a}_i, e_i)$; and let $\ell_0, \ldots, \ell_m$ be such that $\ell_0 = \langle\alpha(\bar{r})\rangle$ and for each $i \in [m]$, $\ell_i = \mathrm{decr}(\ell_{i-1}, \bar{a}_i)$. Notice that each of the $\bar{a}_i$ and ℓ_i can be computed in logarithmic space given $\bar{a}$, m, $e_1, \ldots, e_m$ and i as input. Let $\preceq$ be some fixed ordering on $\bar{a}\mathtt{E}$. There are now two possible cases:

(1) *m was increased in the last move, or there was no last move.* This corresponds to a first visit of the vertex $\bar{a}$ with ℓ on the current path. Therefore we write the sequence $(\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m)$ to the output tape. We then let $j := 0$ be the index of the child of $\bar{a}$ to be visited next.

(2) *m was decreased in the last move.* This corresponds to a return from a child $\bar{b}$ of $\bar{a}$. Therefore, we do not write anything to the output tape. Let $\bar{b}$ be the vertex visited in the last step, let j' be its rank in $\bar{a}\mathtt{E}$ with respect to $\preceq$ (i.e., the number of elements in $\bar{a}\mathtt{E}$ that precede $\bar{b}$ with respect to $\preceq$), and let $j := j' + 1$.

If $\ell > 0$ and $j \leq |\bar{a}\mathtt{E}| - 1$, we update $\bar{a}$ to be the element of rank j in $\bar{a}\mathtt{E}$ with respect to $\preceq$; we also update ℓ to be $\mathrm{decr}(\ell, \bar{a})$, increase m by one, and let e_m be such that $\bar{a}_m = \mathrm{pre}(\bar{a}, e_m)$. Otherwise, if $\ell = 0$ or $j = |\bar{a}\mathtt{E}|$, we do the following. If $m = 0$, we stop; and if $m > 0$ we update $\bar{a}$ to be $\bar{a}_{m-1}$, set ℓ to ℓ_{m-1}, and decrease m by one. It is not hard to see that this procedure outputs all the vertices of T.

Maintaining the vertex $\bar{a} \in \mathtt{V}$ and the vertex from the respective last step needs space $O(\log |V(A)|)$. Notice that

$$\ell_0 = \langle \alpha(\bar{r}) \rangle \leq (|V(A)| + 1)^k - 1.$$

Since $\ell_i = \mathrm{decr}(\ell_{i-1}, \bar{a}_i)$ for every $i \in [m]$, this implies

$$m \leq \ell_0 < (|V(A)| + 1)^k \quad \text{and} \quad \prod_{i=1}^{m} |\mathtt{E}\bar{a}_i| < \frac{\ell_0}{\ell_m} < (|V(A)| + 1)^k. \qquad (3.6)$$

In particular, m together with a bit indicating whether m was increased or decreased in the last move can be maintained in space $O(\log |V(A)|)$. Furthermore, each of the numbers e_i needs space $\eta_i := \lceil \log_2 |\mathtt{E}\bar{a}_i| \rceil$. Let $\mathcal{I}$ be the set of all $i \in [m]$ with $|\mathtt{E}\bar{a}_i| \geq 2$. By (3.6) we have $|\mathcal{I}| \leq \log_2(|V(A)| + 1)^k$. Hence,

$$\sum_{i=1}^{m} \eta_i = \sum_{i \in \mathcal{I}} \lceil \log_2 |\mathtt{E}\bar{a}_i| \rceil \leq |\mathcal{I}| + \log_2 \prod_{i \in \mathcal{I}} |\mathtt{E}\bar{a}_i| \overset{(3.6)}{\leq} 2\log_2(|V(A)| + 1)^k.$$

In particular, we can store $e_1, \ldots, e_m$ as a single number e with $\eta := 2\log_2(|V(A)| + 1)^k - 1)$ bits, reserving η_i bits in e for the number e_i. To extract e_i from e, we start by computing η_m from $\bar{a} = \bar{a}_m$, let e_m be the number represented by the last η_m bits of e, and let $\bar{a}_{m-1} := \mathrm{pre}(\bar{a}_m, e_m)$. We then compute η_{m-1} from $\bar{a}_{m-1}$, let e_{m-1} be the number corresponding to bit η_{m-1} to $\eta - \eta_m$ of e, and let $\bar{a}_{m-2} := \mathrm{pre}(\bar{a}_{m-1}, e_{m-1})$. We continue this way until e_i is found. $\qquad \square$

Claim 3. There is a deterministic logspace Turing machine that takes T as input and decides whether the root $(\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle)$ of T belongs to Y.

Proof. Let $v_0 := (\alpha(\bar{w}), \langle \alpha(\bar{r}) \rangle)$. On input T, a deterministic logspace Turing machine can decide whether $v_0 \in Y$ as follows. The idea is to visit the vertices in a depth-first fashion, starting in v_0, and count, for each node that is visited, the number of children that belong to Y. To implement this in logarithmic space, we proceed in steps as follows.

In each step, we are in a vertex v of T, which is v_0 in the first step. With each vertex v_i on the path $v_0, v_1, \ldots, v_m$ from v_0 to v we associate $2 \cdot \ell_v(i)$ bits of memory for counters $t(i), c(i)$ from 0 to $2^{\ell_v(i)} - 1$, where $\ell_v(i)$ will be specified below. The counter $t(i)$ simply counts the number of children of v_i that have already been processed (excluding the vertex in whose subtree we are currently in), while $c(i)$ counts the number of children of v_i that have already been processed and belong to Y. We guarantee that the sum of the numbers $2 \cdot \ell_v(i)$ over $i \in [0, m]$ is bounded by $6 \cdot \log_2 |W|$. Moreover, it will be easy to determine $\ell_v(i)$ from v and i in logspace; so we can store the counters in a bit string of length at most $6 \cdot \log_2 |W|$, and identify the bits that belong to $t(i)$ and $c(i)$ from that bit string in logspace, given v and i. By visiting the children of each vertex in decreasing order of the number of vertices in the children's subtrees, we ensure that there is always enough space to keep the counters in memory until all children have been processed.

We now give a more detailed description of a single step. In the initial step, we set $v := v_0$ and $t(0) := c(0) := 0$. For the other steps, we need the following definitions:

- The *size* $s(v)$ of a vertex $v \in W$ is the number of vertices in the subtree of T rooted at v. It is easy to compute this number in logarithmic space: all we need to do is to initialise a counter, iterate over all vertices of T, and for each such vertex move upwards and increment the counter by 1 if v is reached.

- Let $v \in W$, and let $w_1, \ldots, w_p$ be the children of v such that $s(w_1) \geq s(w_2) \geq \cdots \geq s(w_p)$; children of the same size are ordered in lexicographic order based on their representation in the input string. For every $j \in [p]$, let $\mathrm{child}(v,j) := w_j$. The vertex $\mathrm{child}(v,j)$ is easy to compute in logarithmic space, given v and j.
- Let $v \in W$, let $v_0, v_1, \ldots, v_m$ be the path from v_0 to v, and let $i \in [0,m]$. Then

$$\ell_v(i) := \begin{cases} \lceil \log_2 j \rceil, & \text{if } i < m \text{ and } \mathrm{child}(v_i, j) = v_{i+1}, \\ \lceil \log_2 |W| \rceil, & \text{if } i = m. \end{cases}$$

This number is easy to compute in logspace given v and i as input.

Suppose that v is the current vertex, and that $v_0, v_1, \ldots, v_m$ is the path from v_0 to v. If $t(m)$ is smaller than the number of children of v, then we set $v := \mathrm{child}(v, t(m) + 1)$ and $t(m+1) := c(m+1) := 0$, and continue with the next step. Otherwise, we check whether $c(m) \in \mathtt{C}(v)$ and $fail(v) = 0$. If this is the case, we say that v *succeeds*. In any case, whether v succeeds or not, we do the following:

(1) If $m = 0$, then we accept T iff v succeeds.
(2) If $m > 0$, then we increase $t(m-1)$ by one, and if v succeeds we also increase $c(m-1)$ by one. Afterwards, we let v be the parent of v, and continue with the next step. Note that with the updated v, $2\ell_v(m-1)$ bits suffice to store $t(m-1)$ and $c(m-1)$.

It should be clear that this procedure correctly decides whether $v_0 \in Y$.

Concerning the space for the counters, let $j_0, j_1, \ldots, j_{m-1}$ be such that $\mathrm{child}(v_i, j_i) = v_{i+1}$ for every $i < m$. Then

$$\sum_{i<m} \ell_v(i) = \sum_{\substack{i<m \\ j_i \geq 2}} \lceil \log_2 j_i \rceil \leq \sum_{\substack{i<m \\ j_i \geq 2}} (1 + \log_2 j_i) = |\{i < m \mid j_i \geq 2\}| + \log_2 \prod_{i<m} j_i. \quad (3.7)$$

Now observe that

$$s(v_{i+1}) < \frac{s(v_i)}{j_i} \qquad \text{for every } i \in [0, m-1]. \quad (3.8)$$

To see this, consider $w_j := \mathrm{child}(v_i, j)$ for every $j \leq j_i$. By the choice of $\mathrm{child}(\cdot, \cdot)$, we have $s(w_1) \geq \cdots \geq s(w_{j_i})$. Hence, if $s(w_{j_i}) = s(v_{i+1}) \geq s(v_i)/j_i$, then $s(w_1) + \cdots + s(w_{j_i}) \geq s(v_i)$, which is impossible. As a consequence of (3.8), we have

$$|\{i < m \mid j_i \geq 2\}| \ < \ \log_2 |W| \qquad \text{and} \qquad \prod_{i<m} j_i \overset{(3.8)}{<} \prod_{i<m} \frac{s(v_i)}{s(v_{i+1})} = \frac{s(v_0)}{s(v_m)} \leq |W|. \quad (3.9)$$

Altogether, this yields

$$\sum_{i \leq m} \ell_v(i) \overset{(3.7)}{\leq} |\{i < m \mid j_i \geq 2\}| + \log_2 \prod_{i<m} j_i + \log_2 |W| + 1 \overset{(3.9)}{<} 3\log_2 |W| + 1,$$

which implies $\sum_{i \leq m} \ell_v(i) \leq 3\log_2 |W|$, and therefore $\sum_{i \leq m} 2\ell_v(i) \leq 6\log_2 |W|$. $\qquad \square$

Altogether, this concludes the proof of Theorem 3.4. $\qquad \blacksquare$

Remark 3.5. It follows from Example 3.2 that $\mathsf{DTC+C} \leq \mathsf{LREC}$. This containment is strict as directed tree isomorphism is definable in LREC (we will show this in the next section), but not in $\mathsf{DTC+C}$. On the other hand, it is easy to see that the relation X defined by an LREC-formula of the form (3.1) in an interpretation (A, α) can be defined in fixed point logic with counting $\mathsf{FP+C}$. Hence, $\mathsf{LREC} \leq \mathsf{FP+C}$, and this containment is strict since we show in Section 5 that undirected graph reachability is not LREC-definable.

4. Capturing Logspace on Directed Trees

In this section we show that LREC captures $\mathsf{LOGSPACE}$ on the class of all directed trees. Our construction is based on Lindell's $\mathsf{LOGSPACE}$ tree canonisation algorithm [23]. Note, however, that Lindell's algorithm makes essential use of a linear order on the tree's vertices that is given implicitly by the encoding of the tree. Here we do not have such a linear order, so we cannot directly translate Lindell's algorithm to an LREC-formula. We show that we can circumvent using the linear order if we have a formula for directed tree isomorphism. Hence, our first task is to construct such a formula.

4.1. **Directed Tree Isomorphism.** Let T be a directed tree. For every $v \in V(T)$ let T_v be the subtree of T rooted at v, let $\text{size}(v) := |V(T_v)|$ be the *size* of v, and let $\#_s(v)$ be the number of children of v of size s. We construct an $\mathsf{LREC}[\{E\}]$-formula $\varphi_{\cong}(x, y)$ that is true in a directed tree T with interpretations $v, w \in V(T)$ for x, y if and only if $T_v \cong T_w$. We assume that $|V(T)| \geq 4$, but it is easy to adapt the construction to directed trees with less than 4 vertices.

We implement the following recursive procedure to check whether $T_v \cong T_w$:

(1) If $\text{size}(v) \neq \text{size}(w)$ or if $\#_s(v) \neq \#_s(w)$ for some $s \in [0, |V(T_v)| - 1]$, then return "$T_v \not\cong T_w$".
(2) If for all children $\hat{v}$ of v there is a child $\hat{w}$ of w and a number k such that
   (a) $T_{\hat{v}} \cong T_{\hat{w}}$,
   (b) there are exactly k children $\mathring{w}$ of w with $T_{\hat{v}} \cong T_{\mathring{w}}$, and
   (c) there are exactly k children $\mathring{v}$ of v with $T_{\mathring{v}} \cong T_{\hat{w}}$,
   then return "$T_v \cong T_w$".
(3) Return "$T_v \not\cong T_w$".

Clearly, this procedure outputs "$T_v \cong T_w$" if and only if $T_v \cong T_w$.

To simplify the presentation we fix a directed tree T and an assignment α in T, but the construction will be uniform in T and α.

We construct a directed graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ with labels $\mathsf{C}(v) \subseteq \mathbb{N}$ for each $v \in \mathsf{V}$ as follows. Let $\mathsf{V} := N(T) \times V(T)^4 \times N(T)$. The first component of each vertex is its *type*; the meaning of the other components will become clear soon. Although G will not be a tree, it is helpful to think of it as a *decision tree* for deciding $T_v \cong T_w$. For each pair $(v, w) \in V(T)^2$, we designate the vertex $\bar{a}_{v,w} = (0, v, w, v, w, 0)$ to stand for "$T_v \cong T_w$". Let us call (v, w) *easy* if v, w satisfy the condition in line 1 of the procedure (i.e., $\text{size}(v) \neq \text{size}(w)$, or $\#_s(v) \neq \#_s(w)$ for some $s \in [0, |V(T_v)| - 1]$). Note that the set of all such easy pairs is LREC-definable.[3] If

---

[3]Using the dtc-operator (3.3) from Example 3.2 we can construct an $\mathsf{LREC}[\{E\}]$-formula defining the descendant relation between vertices in a directed tree, and using this formula it is easy to determine the size and the number of children of size s of a vertex.
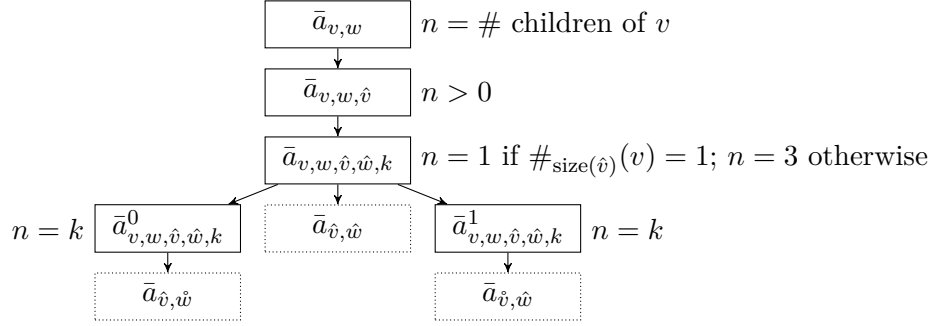
$$\boxed{\bar{a}_{v,w}}\quad n = \#\text{ children of }v$$

$$\boxed{\bar{a}_{v,w,\hat{v}}}\quad n > 0$$

$$\boxed{\bar{a}_{v,w,\hat{v},\hat{w},k}}\quad n = 1 \text{ if } \#_{\text{size}(\hat{v})}(v) = 1;\ n = 3 \text{ otherwise}$$

$$n = k\ \boxed{\bar{a}^0_{v,w,\hat{v},\hat{w},k}} \qquad \boxed{\bar{a}_{\hat{v},\hat{w}}} \qquad \boxed{\bar{a}^1_{v,w,\hat{v},\hat{w},k}}\ n = k$$

$$\boxed{\bar{a}_{\hat{v},\mathring{w}}} \qquad\qquad \boxed{\bar{a}_{\mathring{v},\hat{w}}}$$

Figure 3: Sketch of "decision tree" for deciding $T_v \cong T_w$. Here, $\hat{v}, \mathring{v}$ range over the children of v; $\hat{w}, \mathring{w}$ range over the children of w; and $k \in [\#_{\text{size}(\hat{v})}(v)]$. Moreover, $\hat{v}, \mathring{v}, \hat{w}, \mathring{w}$ all have the same size. Labels indicate which integers n belong to the set $\mathtt{C}(\bar{a})$ labelling each vertex $\bar{a}$. If $\hat{v}$ is the only child of v of size $\text{size}(\hat{v})$, then $\bar{a}_{\hat{v},\hat{w}}$ is the only child of $\bar{a}_{v,w,\hat{v},\hat{w},k}$.

(v, w) is easy, then $\bar{a}_{v,w}$ has no outgoing edges and $\mathtt{C}(\bar{a}_{v,w}) = \emptyset$. On the other hand, if (v, w) is not easy, then $\mathtt{G}$ contains the following edges and labels (see Figure 3 for an illustration):

- The vertex $\bar{a}_{v,w}$ has an outgoing edge to $\bar{a}_{v,w,\hat{v}} := (1, v, w, \hat{v}, w, 0)$, for each child $\hat{v}$ of v. Furthermore, $\mathtt{C}(\bar{a}_{v,w}) = \{\#\text{ of children of }v\}$. This corresponds to "for all children $\hat{v}$ of v..." in the above procedure's step 2.
- The vertex $\bar{a}_{v,w,\hat{v}}$ has an outgoing edge to $\bar{a}_{v,w,\hat{v},\hat{w},k} := (2, v, w, \hat{v}, \hat{w}, k)$, for each child $\hat{w}$ of w with $\text{size}(\hat{w}) = \text{size}(\hat{v})$ and each $k \in [\#_{\text{size}(\hat{v})}(v)]$. Furthermore, $\mathtt{C}(\bar{a}_{v,w,\hat{v}}) = N(T) \setminus \{0\}$. This branching corresponds to "...there is a child $\hat{w}$ of w and a number k such that...".
- The vertex $\bar{a}_{v,w,\hat{v},\hat{w},k}$ has an outgoing edge to $\bar{a}_{\hat{v},\hat{w}}$. If $\hat{v}$ is the only child of v of size $\text{size}(\hat{v})$, then this is the only outgoing edge, and we let $\mathtt{C}(\bar{a}_{v,w,\hat{v},\hat{w},k}) = \{1\}$. Otherwise, there are additional outgoing edges to $\bar{a}^i_{v,w,\hat{v},\hat{w},k} = (3 + i, v, w, \hat{v}, \hat{w}, k)$ for $i \in \{0, 1\}$, and we let $\mathtt{C}(\bar{a}_{v,w,\hat{v},\hat{w},k}) = \{3\}$. This corresponds to conditions 2a–2c.
- The vertex $\bar{a}^0_{v,w,\hat{v},\hat{w},k}$ has outgoing edges to $\bar{a}_{\hat{v},\mathring{w}}$ for each child $\mathring{w}$ of w of size $\text{size}(\hat{v})$, and $\bar{a}^1_{v,w,\hat{v},\hat{w},k}$ has outgoing edges to $\bar{a}_{\mathring{v},\hat{w}}$ for each child $\mathring{v}$ of v of size $\text{size}(\hat{w}) = \text{size}(\hat{v})$. Furthermore, $\mathtt{C}(\bar{a}^i_{v,w,\hat{v},\hat{w},k}) = \{k\}$. The vertex $\bar{a}^i_{v,w,\hat{v},\hat{w},k}$ corresponds to condition 2b for $i = 0$, and to 2c for $i = 1$.

From the above description it should be easy to construct $\mathsf{LREC}[\{E\}]$-formulae $\varphi_{\mathtt{E}}(\bar{u}, \bar{u}')$ and $\varphi_{\mathtt{C}}(\bar{u}, p)$, where $\bar{u} = (q_t, x, y, \hat{x}, \hat{y}, q_k)$ and $\bar{u}' = (q'_t, x', y', \hat{x}', \hat{y}', q'_k)$, such that $\varphi_{\mathtt{E}}[T, \alpha; \bar{u}, \bar{u}'] = \mathtt{E}$, and $\{\langle n\rangle \mid n \in \varphi_{\mathtt{C}}[T, \alpha[\bar{a}/\bar{u}]; p]\} = \mathtt{C}(\bar{a})$ for each $\bar{a} \in \mathtt{V}$.

Let

$$\varphi_{\cong}(x, y) := \exists \bar{r}\, [\mathsf{lrec}_{\bar{u},\bar{u}',p}\, \varphi_{\mathtt{E}},\ \varphi_{\mathtt{C}}]((0, x, y, x, y, 0), \bar{r}),$$

where $\bar{r}$ is a 5-tuple of number variables.[4] Let X be the relation defined by $\varphi_{\cong}$ in (T, α). Then:

Lemma 4.1. *Let $v, w \in V(T)$.*
(1) *If $(\bar{a}_{v,w}, \ell) \in X$ for some $\ell \in \mathbb{N}$, then $T_v \cong T_w$.*
(2) *If $T_v \cong T_w$, then for all $\ell \geq \text{size}(v)^5$ we have $(\bar{a}_{v,w}, \ell) \in X$.*

---

[4]We use 0 as a constant, but clearly we can modify $\varphi_{\cong}$ to a formula that does not use the constant 0.

Proof. Ad (1): The proof is by induction on $\mathrm{size}(v)$. If $\mathrm{size}(v) = 1$ and $(\bar{a}_{v,w}, \ell) \in X$, then (v, w) is not easy, which implies $\mathrm{size}(w) = 1$ and hence $T_v \cong T_w$.

Now let $\mathrm{size}(v) = s+1$ for some $s \geq 1$. If $(\bar{a}_{v,w}, \ell) \in X$, then (v, w) is not easy, implying $\mathrm{size}(w) = s + 1$ and $\#_t(v) = \#_t(w)$ for all $t \in \mathbb{N}$. It is then easy to see that for all children $\hat{v}$ of v in T there is a child $\hat{w}$ of w in T and a number $k \in [1, \#_{\mathrm{size}(\hat{v})}v]$ such that

- $(\bar{a}_{\hat{v},\hat{w}}, \ell') \in X$ for some $\ell' \in \mathbb{N}$,
- there are exactly k children $\mathring{w}$ of w such that $(\bar{a}_{\hat{v},\mathring{w}}, \ell') \in X$ for some $\ell' \in \mathbb{N}$, and
- there are exactly k children $\mathring{v}$ of v such that $(\bar{a}_{\mathring{v},\hat{w}}, \ell') \in X$ for some $\ell' \in \mathbb{N}$.

By the induction hypothesis, this corresponds to step 2 of the procedure given at the beginning of Section 4.1, and therefore implies $T_v \cong T_w$.

Ad (2): The proof is by induction on $\mathrm{size}(v)$. Suppose that $\mathrm{size}(v) = 1$ and $T_v \cong T_w$. Then $\mathrm{size}(w) = 1$ which implies that (v, w) is not easy. Furthermore, as v has no children in T, we know that $\bar{a}_{v,w}$ has no children in G and $\mathsf{C}(\bar{a}_{v,w}) = \{0\}$. Hence, $(\bar{a}_{v,w}, \ell) \in X$ for all $\ell \geq 1 = \mathrm{size}(v)^5$.

Now suppose that $\mathrm{size}(v) = s + 1$ for some $s \geq 1$, and $T_v \cong T_w$. First note that (v, w) is not easy. Let $\ell \geq (s+1)^5$. We show that $(\bar{a}_{v,w,\hat{v}}, \ell - 1) \in X$ for all children $\hat{v}$ of v, which implies $(\bar{a}_{v,w}, \ell) \in X$. Let $\hat{v}$ be a child of v in T. Since $T_v \cong T_w$, there is a child $\hat{w}$ of w of size $s' := \mathrm{size}(\hat{v})$ and a number $k \in [\#_{s'}(v)]$ such that

- $T_{\hat{v}} \cong T_{\hat{w}}$,
- there are exactly k children $\mathring{w}$ of w of size s' such that $T_{\hat{v}} \cong T_{\mathring{w}}$, and
- there are exactly k children $\mathring{v}$ of v of size s' such that $T_{\mathring{v}} \cong T_{\hat{w}}$.

Pick such $\hat{w}$ and k.

Let us deal with the case $\#_{s'}(v) = 1$ first. In this case, $\bar{a}_{\hat{v},\hat{w}}$ is the only child of $\bar{a}_{v,w,\hat{v},\hat{w},k}$; moreover, $\bar{a}_{v,w,\hat{v},\hat{w},k}$ and $\bar{a}_{\hat{v},\hat{w}}$ have exactly one incoming edge each. Since $T_{\hat{v}} \cong T_{\hat{w}}$ and $\ell - 3 \geq (s')^5$, the induction hypothesis implies $(\bar{a}_{\hat{v},\hat{w}}, \ell - 3) \in X$. Consequently $(\bar{a}_{v,w,\hat{v}}, \ell - 1) \in X$.

In the following we assume $\#_{s'}(v) \geq 2$. Let $d := 3 \cdot \#_{s'}(v)^2$. Note that all vertices in Figure 3 except the type 0-vertices have exactly one incoming edge, and that the in-degree d' of a type 0-vertex $\bar{a}_{v',w'}$, where v', w' are children of v and w, respectively, of size s' is at most d, because it has incoming edges from

- vertices $\bar{a}_{v,w,v',w',k}$, where v and w are the (unique) parents of v' and w', respectively, and $k \in [\#_{s'}(v)]$;
- vertices $\bar{a}^0_{v,w,v',\hat{w},k}$, where v, w, k are as above and $\hat{w}$ is a child of w of size s'; and
- vertices $\bar{a}^1_{v,w,\hat{v},w',k}$, where v, w, k are as above and $\hat{v}$ is a child of v of size s'.

Let $\ell' := \lfloor (\ell - 4)/d \rfloor$. Then

$$\ell' \geq \frac{\ell - d - 3}{d} \geq \frac{s^5}{d} + \frac{s^4}{d} - 2 \geq \frac{\#_{s'}(v)^5 \cdot (s')^5}{3 \cdot \#_{s'}(v)^2} + \frac{\#_{s'}(v)^4}{3 \cdot \#_{s'}(v)^2} - 2 \geq 2(s')^5 - 1 \geq (s')^5,$$

where for the second inequality we use $(s+1)^5 \geq s^5 + s^4$, for the third one we use $\#_{s'}(v) \cdot s' \leq s$, and for the fourth one we use $\#_{s'}(v) \geq 2$. Hence, by the induction hypothesis we have:

- $(\bar{a}_{\hat{v},\hat{w}}, \lfloor (\ell - 3)/d' \rfloor) \in X$ (note that $\lfloor (\ell - 3)/d' \rfloor \geq \ell'$).
- There are exactly k children $\mathring{w}$ of w of size s' with $(\bar{a}_{\hat{v},\mathring{w}}, \lfloor (\ell - 4)/d' \rfloor) \in X$ (note that $\lfloor (\ell - 4)/d' \rfloor \geq \ell'$), which implies $(\bar{a}^0_{v,w,\hat{v},\hat{w},k}, \ell - 3) \in X$.
- There are exactly k children $\mathring{v}$ of v of size s' with $(\bar{a}_{\mathring{v},\hat{w}}, \lfloor (\ell - 4)/d' \rfloor) \in X$, which implies that $(\bar{a}^1_{v,w,\hat{v},\hat{w},k}, \ell - 3) \in X$.

It follows immediately that $(\bar{a}_{v,w,\hat{v},\hat{w},k}, \ell - 2) \in X$, and therefore $(\bar{a}_{v,w,\hat{v}}, \ell - 1) \in X$. $\quad\square$

Corollary 4.2. *Let* $v, w \in V(T)^2$. *Then,* $T \models \varphi_{\cong}[v, w]$ *if and only if* $T_v \cong T_w$.

Proof. $T \models \varphi_{\cong}[v, w]$ holds precisely when $(\bar{a}_{v,w}, |N(T)|^{|\bar{r}|} - 1) \in X$. Furthermore, $|N(T)|^{|\bar{r}|} - 1 \geq |V(T)|^5 \geq \operatorname{size}(v)^5$. Therefore, by the preceding lemma, $(\bar{a}_{v,w}, |N(T)|^{|\bar{r}|} - 1) \in X$ is equivalent to $T_v \cong T_w$, and the claim follows. $\quad\square$

4.2. **Defining an Order on Directed Trees.** Lindell's tree canonisation algorithm is based on a logspace-computable linear order on isomorphism classes of directed trees. We show that a slightly refined version of this order is LREC-definable.

Let T be a directed tree. For each $v \in V(T)$ let $\pi(v) := (\operatorname{size}(v), \#_1(v), \ldots, \#_{\operatorname{size}(v)-1}(v))$ be the *profile* of v.[5] Let $\preceq$ be the total preorder on $V(T)$,[6] where $v \prec w$ whenever

(1) $\pi(v) < \pi(w)$ lexicographically, or

(2) $\pi(v) = \pi(w)$ and the following is true: Let $v_1, \ldots, v_k$ and $w_1, \ldots, w_k$ be the children of v and w, respectively, ordered such that $v_1 \preceq \cdots \preceq v_k$ and $w_1 \preceq \cdots \preceq w_k$. Then there is an $i \in [k]$ with $v_i \prec w_i$, and for all $j < i$ we have $v_j \preceq w_j$ and $w_j \preceq v_j$.

Note that $v \preceq w$ and $w \preceq v$ imply $T_v \cong T_w$. We show that $\preceq$ is LREC-definable.

To simplify the presentation, we again fix a directed tree T and an assignment α, and we assume that $|V(T)| \geq 4$.

We apply the lrec-operator to the following graph $\mathtt{G} = (\mathtt{V}, \mathtt{E})$ with labels $\mathtt{C}(v) \subseteq \mathbb{N}$ for each $v \in \mathtt{V}$. Let $\mathtt{V} := N(T) \times V(T)^4 \times N(T)$. For each $(v, w) \in V(T)^2$, the vertex $\bar{a}_{v,w} = (0, v, w, v, w, 0)$ represents "$v \prec w$". If $\pi(v) < \pi(w)$, then $\bar{a}_{v,w}$ has no outgoing edges and $\mathtt{C}(\bar{a}_{v,w}) = \{0\}$. If $\pi(v) > \pi(w)$, then $\bar{a}_{v,w}$ has no outgoing edges and $\mathtt{C}(\bar{a}_{v,w}) = \emptyset$. Note that the relation "$\pi(v) \leq \pi(w)$" is LREC-definable.

Suppose that $\pi(v) = \pi(w)$. For all $t, u \in V(T)$ let $\theta_u(t)$ be the number of children u' of u with $T_{u'} \cong T_t$. Call a child $\hat{v}$ of v *good* if $\theta_v(\hat{v}) > \theta_w(\hat{v})$ and for all children v' of v with $\operatorname{size}(v') < \operatorname{size}(\hat{v})$ we have $\theta_v(v') = \theta_w(v')$. Then it is not hard to see that $v \prec w$ precisely if there is a good child $\hat{v}$ of v, a child $\hat{w}$ of w of size $s := \operatorname{size}(\hat{v})$ and a $k \in [\#_s(v)]$ such that:

- $\hat{v} \prec \hat{w}$;
- there are exactly k children $\mathring{w}$ of w of size s with $\mathring{w} \prec \hat{v}$;
- there are exactly k children $\mathring{v}$ of v of size s with $\mathring{v} \prec \hat{w}$ and $T_{\mathring{v}} \not\cong T_{\hat{v}}$;
- and for all k children w' of w of size s with $w' \prec \hat{v}$ we have $\theta_v(w') = \theta_w(w')$.

The "decision tree" in Figure 4 checks precisely these conditions.

Using the formula $\varphi_{\cong}$ from the previous section it is now straightforward to construct LREC$[\{E\}]$-formulae $\varphi_{\mathtt{E}}(\bar{u}, \bar{u}')$ and $\varphi_{\mathtt{C}}(\bar{u}, p)$ that define the edge relation $\mathtt{E}$ of $\mathtt{G}$ and the sets $\mathtt{C}(\bar{a})$ for each $\bar{a} \in \mathtt{V}$, where $\bar{u}$ and $\bar{u}'$ are as in the definition of $\varphi_{\cong}$. Let

$$\varphi_{\prec}(x, y) := \exists \bar{r} \, [\mathsf{lrec}_{\bar{u}, \bar{u}', p} \, \varphi_{\mathtt{E}}, \, \varphi_{\mathtt{C}}]((0, x, y, x, y, 0), \bar{r}),$$

where $\bar{r}$ is a 5-tuple of number variables. Let X be the relation defined by $\varphi_{\prec}$ in (T, α). We then have:

Lemma 4.3. *Let* $v, w \in V(T)$.

(1) *If* $(\bar{a}_{v,w}, \ell) \in X$ *for some* $\ell \in \mathbb{N}$, *then* $v \prec w$.

---

[5]Lindell's order can be obtained by replacing $\pi(v)$ with $\pi'(v) := (\operatorname{size}(v), \#\text{children of } v)$.

[6]That is, $\preceq$ is a preorder on $V(T)$ such that for all $v, w \in V(T)$ we have $v \preceq w$ or $w \preceq v$.
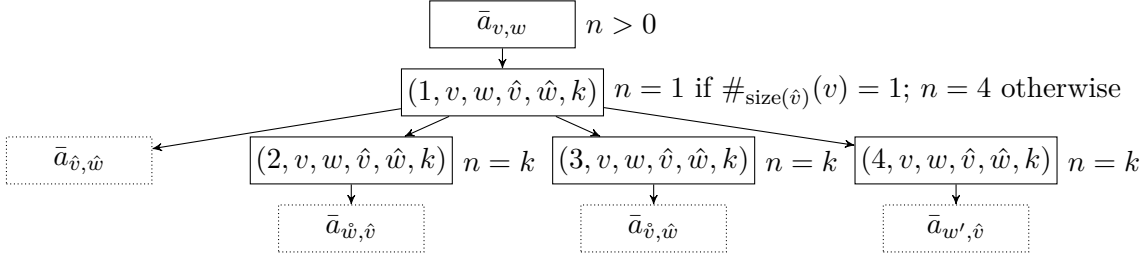
Figure 4: Gadget for deciding $v \prec w$ when $\pi(v) = \pi(w)$. Here, $\hat{v}$ ranges over good children of v; $\mathring{v}$ ranges over children of v of size $s := \text{size}(v)$ and $T_{\mathring{v}} \not\cong T_{\hat{v}}$; $\hat{w}, \mathring{w}$ range over children of w of size s; w' ranges over children of w of size s with $\theta_v(w') = \theta_w(w')$; and $k \in [\#_s(v)]$. The edges from $(2, v, w, \hat{v}, \hat{w}, k)$ to $(t, \dots)$ for $t \in \{2, 3, 4\}$ exist only if $\#_s(v) > 1$. Labels indicate which integers n belong to the set $\mathtt{C}(\bar{a})$ labelling each vertex $\bar{a}$.

(2) *If $v \prec w$, then for all $\ell \geq \text{size}(v)^5$ we have $(\bar{a}_{v,w}, \ell) \in X$.*

Proof. The proof is similar to the proof of Lemma 4.1.

Ad (1): The proof is by induction on $\text{size}(v)$. Suppose $\text{size}(v) = 1$. If $(\bar{a}_{v,w}, \ell) \in X$, then $\pi(v) \leq \pi(w)$. We cannot have $\pi(v) = \pi(w)$, since otherwise $0 \notin \mathtt{C}(\bar{a}_{v,w})$ (see Figure 4), so that X would contain at least one tuple of the form $((1, v, w, \hat{v}, \cdot, \cdot), \ell - 1)$ with $\hat{v}$ a child of v. But such a tuple does not exist, since v has no children. It follows that $\pi(v) < \pi(w)$ which implies $v \prec w$.

Now let $\text{size}(v) = s + 1$ for some $s \geq 1$. If $(\bar{a}_{v,w}, \ell) \in X$, then as above we have $\pi(v) \leq \pi(w)$. If $\pi(v) < \pi(w)$, we have $v \prec w$. So, suppose that $\pi(v) = \pi(w)$, that is, $\text{size}(w) = s + 1$ and $\#_t(v) = \#_t(w)$ for all $t \in \mathbb{N}$. It is then easy to see that there is a good child $\hat{v}$ of v, a child $\hat{w}$ of w of size $s := \text{size}(\hat{v})$, and a $k \in [\#_s(v)]$ such that

- $(\bar{a}_{\hat{v}, \hat{w}}, \ell') \in X$ for some $\ell' \in \mathbb{N}$,
- there are exactly k children $\mathring{w}$ of w of size s such that $(\bar{a}_{\mathring{w}, \hat{v}}, \ell') \in X$ for some $\ell' \in \mathbb{N}$,
- there are exactly k children $\mathring{v}$ of v of size s with $T_{\mathring{v}} \not\cong T_{\hat{v}}$ such that $(\bar{a}_{\mathring{v}, \hat{w}}, \ell') \in X$ for some $\ell' \in \mathbb{N}$, and
- all k children w' of w of size s with $(\bar{a}_{\mathring{w}, \hat{v}}, \ell') \in X$ for some $\ell' \in \mathbb{N}$ satisfy $\theta_v(w') = \theta_w(w')$.

By the induction hypothesis, this means that

- $\hat{v} \prec \hat{w}$,
- there are exactly k children $\mathring{w}$ of w of size s such that $\mathring{w} \prec \hat{v}$,
- there are exactly k children $\mathring{v}$ of v of size s with $T_{\mathring{v}} \not\cong T_{\hat{v}}$ such that $\mathring{v} \prec \hat{w}$, and
- all k children w' of w of size s with $w' \prec \hat{v}$ satisfy $\theta_v(w') = \theta_w(w')$.

As pointed out in Section 4.2, this implies $v \prec w$.

Ad (2): The proof is by induction on $\text{size}(v)$. If $\text{size}(v) = 1$ and $v \prec w$, then $\pi(v) < \pi(w)$. By the construction of $\mathtt{G}$ this immediately implies $(\bar{a}_{v,w}, \ell) \in X$ for all $\ell \geq 1 = \text{size}(v)^5$.

Now suppose that $\text{size}(v) = s + 1$ for some $s \geq 1$, and $v \prec w$. First note that $\pi(v) \leq \pi(w)$. If $\pi(v) < \pi(w)$, then $(\bar{a}_{v,w}, \ell) \in X$ for all $\ell \geq 1$, and in particular, for all $\ell \geq \text{size}(v)^5$. So, assume that $\pi(v) = \pi(w)$.

Since $v \prec w$, there is a good child $\hat{v}$ of v, a child $\hat{w}$ of w of size $s' := \text{size}(\hat{v})$ and a $k \in [\#_{s'}(v)]$ such that

- $\hat{v} \prec \hat{w}$,
- there are exactly k children $\mathring{w}$ of w of size s' with $\mathring{w} \prec \hat{v}$,
- there are exactly k children $\mathring{v}$ of v of size s' with $\mathring{v} \prec \hat{w}$ and $T_{\mathring{v}} \not\cong T_{\hat{v}}$, and
- for all k children w' of w of size s' with $w' \prec \hat{v}$ we have $\theta_v(w') = \theta_w(w')$.

Pick such $\hat{v}$, $\hat{w}$ and k.

If $\#_{s'}(v) = 1$, then $\bar{a}_{\hat{v},\hat{w}}$ is the only child of $(1, v, w, \hat{v}, \hat{w}, k)$, and $(1, v, w, \hat{v}, \hat{w}, k)$ and $\bar{a}_{\hat{v},\hat{w}}$ each have exactly one incoming edge. Since $\hat{v} \prec \hat{w}$ and $\ell - 2 \geq (s')^5$, the induction hypothesis implies $(\bar{a}_{\hat{v},\hat{w}}, \ell - 2) \in X$, and consequently, $(\bar{a}_{v,w}, \ell) \in X$.

In the following we assume $\#_{s'}(v) \geq 2$. Let $d := 4\#_{s'}(v)^2$. Note that all vertices in Figure 4 except the type 0-vertices have exactly one incoming edge. The type 0-vertices $\bar{a}_{v',w'}$, where v', w' are children of v and w, respectively, of size s', have incoming edges from

- vertices $(1, v, w, v', w', k)$, where $k \in [\#_{s'}(v)]$;
- vertices $(2, w, v, w', v'', k)$, where k is as above and v'' is a child of v of size s';
- vertices $(3, v, w, v'', w', k)$, where k and v'' is a good child of v; and
- vertices $(4, w, v, w', v'', k)$, where k and v'' is a child of v of size s'.

Hence, the in-degree of $\bar{a}_{v',w'}$ is at most d. For the type 0-vertices $\bar{a}_{w',v'}$, where v', w' are children of v and w, respectively, of size s', this is symmetric.

Let $\ell \geq (s+1)^5$ and $\ell' := \lfloor(\ell - 3)/d\rfloor$. Then

$$\ell' \geq \frac{\ell - d - 2}{d} \geq \frac{s^5}{d} + \frac{s^4}{d} - 2 \geq \frac{\#_{s'}(v)^5 \cdot (s')^5}{4 \cdot \#_{s'}(v)^2} + \frac{\#_{s'}(v)^4}{4 \cdot \#_{s'}(v)^2} - 2 \geq 2(s')^5 - 1 \geq (s')^5,$$

where for the second inequality we use $(s+1)^5 \geq s^5 + s^4$, for the third one we use $\#_{s'}(v) \cdot s' \leq s$, and for the fourth one we use $\#_{s'}(v) \geq 2$ Hence, by the induction hypothesis we have:

- $(\bar{a}_{\hat{v},\hat{w}}, \lfloor(\ell - 2)/d_1\rfloor) \in X$, where $d_1 \leq d$ is the in-degree of $\bar{a}_{\hat{v},\hat{w}}$,
- there are exactly k children $\mathring{w}$ of w of size s' with $(\bar{a}_{\mathring{w},\hat{v}}, \lfloor(\ell - 3)/d_2\rfloor) \in X$, where $d_2 \leq d$ is the in-degree of the vertices $\bar{a}_{\mathring{w},\hat{v}}$,
- there are exactly k children $\mathring{v}$ of v of size s' with $T_{\mathring{v}} \not\cong T_{\hat{v}}$ and $(\bar{a}_{\mathring{v},\hat{w}}, \lfloor(\ell - 3)/d_3\rfloor) \in X$, where $d_3 \leq d$ is the in-degree of the vertices $\bar{a}_{\mathring{v},\hat{w}}$, and
- for all k children w' of w of size s' with $(\bar{a}_{w',\hat{v}}, \lfloor(\ell - 3)/d_2\rfloor) \in X$ we have $\theta_v(w') = \theta_w(w')$.

It follows immediately that $(\bar{a}_{v,w}, \ell) \in X$. □

Corollary 4.4. *Let $v, w \in V(T)$. Then, $T \models \varphi_\prec[v, w]$ if and only if $v \prec w$.*

4.3. **Canonising Directed Trees.** We now construct an LREC-formula $\gamma(p, q)$ such that for every directed tree T we have $T \cong ([|V(T)|], \gamma[T; p, q])$. Since DTC captures LOGSPACE on ordered structures [15] and a linear order is available on the number sort, we immediately obtain:

Theorem 4.5. LREC *captures* LOGSPACE *on the class of directed trees.*

Since directed tree isomorphism is in LOGSPACE by Lindell's tree canonisation algorithm, but not TC+C-definable [6], we obtain:

Corollary 4.6. LREC $\not\leq$ TC+C *on the class of all directed trees.*

We use l-recursion to define a set $X \subseteq V(T) \times N(T)^2$ (for simplicity, we omit the "resources" in the description) such that for every $v \in V(T)$ the set $X_v := \{(m, n) \in N(T)^2 \mid (v, m, n) \in X\}$ is the edge relation of an isomorphic copy $([|V(T_v)|], X_v)$ of T_v.

Each vertex of T is numbered by its position in the preorder traversal sequence, e.g., the root is numbered 1, its first child v_1 is numbered 2, its second child v_2 is numbered $2+\text{size}(v_1)$, and so on.

To apply the lrec operator, we define a graph $G = (V, E)$ with labels $C(v) \subseteq \mathbb{N}$ for each $v \in V$ as follows. Let $V := V(T) \times N(T)^2$, where $(v, m, n) \in V$ stands for "$(m, n) \in X_v$?". If v is a leaf, then X_v should be empty, so for all $m, n \in N(T)$ we let (v, m, n) have no outgoing edges and define $C((v, m, n)) := \emptyset$. Suppose that v is not a leaf and w is a child of v. Let D_w be the set of all children w' of v with $w' \prec w$, and let e_w be the number of children w' of v with $T_w \cong T_{w'}$. For each $i \in [0, e_w - 1]$, the set X_v will contain an edge from 1 to $p_{w,i} := 2 + \sum_{w' \in D_w} \text{size}(w') + i \cdot \text{size}(w)$, and the edges in $\{(p_{w,i} - 1 + m, p_{w,i} - 1 + n) \mid (m, n) \in X_w\}$. Hence we let $(v, 1, p_{w,i})$ have no outgoing edges and define $C((v, 1, p_{w,i})) := \{0\}$. Furthermore, for all $m, n \in N(T)$ and all $i < e_w$, we let $\bar{a} := (v, p_{w,i} - 1 + m, p_{w,i} - 1 + n)$ have an edge to (w, m, n) and define $C(\bar{a}) := \{e_w\}$.

It is now easy to construct LREC-formulae $\varphi_E(x_1, p_1, p_1', x_2, p_2, p_2')$ and $\varphi_C(x_1, p_1, p_1', q)$ that define the graph G and the labels $C(\cdot)$. Let

$$\gamma(p_1, p_2) := \exists x \exists r (\text{``}x \text{ is the root''} \wedge [\text{lrec}_{(x_1, p_1, p_1'), (x_2, p_2, p_2'), q} \ \varphi_E, \ \varphi_C]((x, p_1, p_2), r)).$$

Noting that the in-degree of each vertex (v, m, n) is at most e_v, it is straightforward to show that γ defines an isomorphic copy of a directed tree:

Lemma 4.7. *Let X be the relation defined by γ in T, let $v \in V(T)$ and let $X_v := \{(m, n) \mid ((v, m, n), \ell) \in X \text{ for some } \ell \geq \text{size}(v)\}$. Then $T_v \cong ([|V(T_v)|], X_v)$.*

Proof. The proof is by induction on $\text{size}(v)$. Clearly, the lemma is true if $\text{size}(v) = 1$. Suppose that $\text{size}(v) = s + 1$. By the induction hypothesis, for each child w of v we have $T_w \cong ([|V(T_w)|], X_w)$.

Let $\ell \geq \text{size}(v)$. Since for all children w of v and all $m, n \in N(T)$, the in-degree of (w, m, n) in G is at most e_w and $e_w \cdot \text{size}(w) < \text{size}(v)$ (which implies $\lfloor (\ell - 1)/e_w \rfloor \geq \lfloor (\text{size}(v) - 1)/e_w \rfloor \geq \text{size}(w)$),

$$\{(p_{w,i} - 1 + m, p_{w,i} - 1 + m) \mid (m, n) \in X_w\} \subseteq X_v \quad \text{for each child } w \text{ of } v \text{ and } i < e_w.$$

Furthermore, by construction, we have $(1, p_{w,i}) \in X_v$ for each child w of v and $i < e_w$, and there are no more edges. It is easy to see that $T_v \cong ([|V(T_v)|], X_v)$. $\square$

Remark 4.8. The results of this section extend to *coloured directed trees* with a linear order on the colours. To be precise, consider a directed tree T and a total preorder $\trianglelefteq$ on $V(T)$. Let $\preceq$ be as in Section 4.2. We define a refinement $\preceq'$ of $\preceq$ by letting $v \prec' w$ whenever $v \lhd w$, or: $v \trianglelefteq w$ and $w \trianglelefteq v$ and $v \prec w$. It should be obvious how to modify $\varphi_\prec(x, y)$ to an LREC$[\{E, \trianglelefteq\}]$-formula $\varphi_\prec'(x, y)$ defining $\prec'$. Using this formula, we then obtain a formula $\gamma'(p, q)$ such that $(V(T), E(T), \trianglelefteq) \cong ([|V(T)|], \gamma'[T; p, q], \trianglelefteq')$, where $m \trianglelefteq' n$ iff for the vertices v, w that correspond to m, n we have $v \trianglelefteq w$.

5. Inexpressibility of Reachability in Undirected Graphs

While LREC captures LOGSPACE on directed trees, its expressive power still lacks the ability to define certain important problems on undirected graphs that can be defined easily in other logics such as STC with LOGSPACE data complexity. As an example, we show in this section that LREC cannot define reachability in undirected graphs:
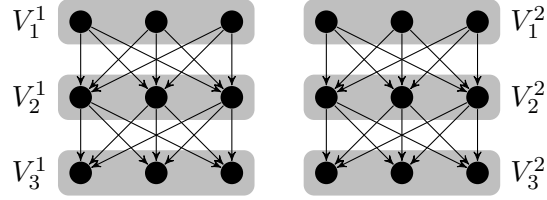
Figure 5: The graph G_3. The gray areas highlight the different layers of G_3.

Theorem 5.1. *There is no* LREC$[\{E\}]$*-formula* $\varphi(x, y)$ *such that for all undirected graphs* G *and all* $v, w \in V(G)$, $G \models \varphi[v, w]$ *iff there is a path from* v *to* w *in* G.

As an immediate corollary we obtain:

Corollary 5.2. STC $\not\leq$ LREC

To prove Theorem 5.1, we show that reachability is not LREC-definable on a certain class of directed graphs. This class, called $\mathcal{C}$ throughout this section, is defined in terms of the following family of graphs G_n, for $n \geq 1$. Here, each graph G_n consists of $2 \cdot n^2$ vertices, which are partitioned into *layers* $V_1^1, \ldots, V_n^1, V_1^2, \ldots, V_n^2$ with $|V_i^j| = n$. Any two vertices in consecutive layers V_i^j and V_{i+1}^j are connected by an edge. That is, the set $E(G_n)$ of edges of G_n is $\{(v, w) \in V_i^j \times V_{i+1}^j \mid i \in [n-1], j \in [2]\}$. For example, the graph G_3 is shown in Figure 5. Now, the class $\mathcal{C}$ is defined as:

$$\mathcal{C} := \{G \mid G \text{ is a graph such that } G \cong G_n \text{ for some } n \geq 1\}.$$

The key property of the graphs in $\mathcal{C}$ that enables us to show that reachability on $\mathcal{C}$ is not LREC-definable is that they are rich in a certain kind of automorphisms. Indeed, let v and w be nodes occurring in the same layer of G_n. Then there is an automorphism of G_n swapping v and w, and fixing the remaining vertices point-wise. To see why this could be useful at all, consider an LREC-formula φ of the form $[\mathsf{lrec}_{\bar{u}_1, \bar{u}_2, \bar{p}} \, \varphi_\mathsf{E}, \, \varphi_\mathsf{C}](\bar{w}, \bar{r})$, and suppose we want to decide membership of a tuple $(\bar{a}_0, \ell_0)$ in the relation X defined by φ in (G_n, α), for an assignment α. First, we would compute the graph G with vertex set $G_n^{\bar{u}_1}$ and edge set E defined by φ_E, and then we would recurse to decide which of the tuples $(\bar{a}_1, \ell_1)$, for successor nodes $\bar{a}_1$ of $\bar{a}_0$ in G and $\ell_1 = \lfloor (\ell_0 - 1)/|\mathsf{E}\bar{a}_1| \rfloor$, belong to X. To decide membership of each of the tuples $(\bar{a}_1, \ell_1)$ in X, we again have to recurse to decide which of the tuples $(\bar{a}_2, \ell_2)$, for successor nodes $\bar{a}_2$ of $\bar{a}_1$ in G and $\ell_2 = \lfloor (\ell_1 - 1)/|\mathsf{E}\bar{a}_2| \rfloor$, belong to X, and so on. Exploiting the above-mentioned automorphisms enables us to show that along each branch $(\bar{a}_0, \ell_0), (\bar{a}_1, \ell_1), (\bar{a}_2, \ell_2), \ldots$ of the "recursion tree", we see only a constant number of tuples $(\bar{a}_{i+1}, \ell_{i+1})$, where $\bar{a}_{i+1}$ does not contain all the vertices of G_n that occur in $\bar{a}_i$, or vice versa. Thus, we are left with finitely many sub-branches "in between" those tuples that contain the same vertices of G_n. If all those sub-branches had constant length, then the whole "recursion tree" would have constant depth, so that we could easily find an FO+C-formula that is equivalent to φ on $\mathcal{C}$ (provided φ_E and φ_C are equivalent to FO+C-formulae). Since reachability is not FO+C definable on $\mathcal{C}$, this would immediately imply Theorem 5.1. In general, the sub-branches do not have constant length (due to number variables that may occur in $\bar{u}_1$ and $\bar{u}_2$), so that we move to a logic that is more expressive than FO+C, but still lacks the ability to define reachability on $\mathcal{C}$.

More precisely, we show that on $\mathcal{C}$, every LREC$[\{E\}]$-formula is equivalent to a formula in the *infinitary counting logic* $\mathcal{L}_{\infty\omega}^*(\mathbf{C})$, introduced in [21] (see also [22, Section 8.2]). The

fact that $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$-formulae without free number variables are Gaifman-local [21] then yields that reachability is not $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$-definable, and hence not LREC-definable, on $\mathcal{C}$.

5.1. **The Logic $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$.** Before delving into the details of translating LREC-formulae into $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$-formulae, we give here a brief review of the logic $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$. For a detailed account, we refer the reader to [21], or [22, Section 8.2].

$\mathcal{L}^*_{\infty\omega}(\mathbf{C})$ on the one hand extends FO+C by allowing for infinite disjunctions and conjunctions, and on the other hand imposes restrictions so as to make the resulting logic not too powerful. While in the context of FO+C, we equipped structures A with a counting sort $N(A) = [0, |V(A)|]$, in the context of $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$ we extend this counting sort to the set of all natural numbers. Furthermore, $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$-formulae may use any natural number $n \in \mathbb{N}$ as a constant, which is always interpreted as n.

$\mathcal{L}^*_{\infty\omega}(\mathbf{C})$ is a restriction of the extremely powerful logic $\mathcal{L}_{\infty\omega}(\mathbf{C})$, which is defined as follows. A *term* t is a structure variable, a number variable, or a non-negative integer; if t is a structure variable, we call t *structure term*, and otherwise *number term*. The atomic formulae of $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$ have the form $R(x_1, \ldots, x_r)$, where $R \in \tau$, r is the arity of R, and $x_1, \ldots, x_r$ are structure variables; or $t = u$, where t and u are either structure terms or number terms; or $t \le u$, where t and u are number terms. The set of all $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$-formulae is the smallest set that contains all atomic formulae, and is closed under the following formula formation rules:

(1) If $\varphi \in \mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$, then $\neg\varphi \in \mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$.
(2) If $\Phi \subseteq \mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$, then $\bigvee\Phi$ and $\bigwedge\Phi$ belong to $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$.
(3) If $\varphi \in \mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$ and x is a variable, then $\exists x\varphi$ and $\forall x\varphi$ belong to $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$.
(4) If $\varphi \in \mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$, x is a structure variable, and $n \in \mathbb{N}$, then $\exists^{\ge n}x\varphi \in \mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$.
(5) If $\varphi \in \mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$, $\bar{x}$ is a tuple of structure variables, and $\bar{p}$ is a tuple of number terms, then $\#\bar{x}\,\varphi = \bar{p}$ belongs to $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$.

Note that, in contrast to FO+C, $\mathcal{L}_{\infty\omega}(\mathbf{C})$ restricts us to tuples of structure variables in counting formulae $\#\bar{x}\,\varphi = \bar{p}$. The semantics of $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$-formulae constructed as in 1, 3, and 5 is as usual. The semantics of formulae of the form $\bigvee\Phi$ or $\bigwedge\Phi$ is "at least one $\varphi \in \Phi$ is satisfied" and "all $\varphi \in \Phi$ are satisfied", respectively. Formulae of the form $\exists^{\ge n}x\varphi$ have the meaning "there are at least n assignments to x for which φ is satisfied".

$\mathcal{L}^*_{\infty\omega}(\mathbf{C})[\tau]$-formulae are those $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$-formulae whose *rank* is bounded. Here, the *rank* $\mathrm{rk}(\varphi)$ of a $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$-formula φ is defined as follows. For atomic formulae φ we have $\mathrm{rk}(\varphi) = 0$. Furthermore, $\mathrm{rk}(\neg\varphi) = \mathrm{rk}(\varphi)$, $\mathrm{rk}(\bigvee\Phi) = \mathrm{rk}(\bigwedge\Phi) = \sup_{\varphi\in\Phi}\mathrm{rk}(\varphi)$, $\mathrm{rk}(\exists x\varphi) = \mathrm{rk}(\forall x\varphi) = \mathrm{rk}(\exists^{\ge n}x\varphi) = 1 + \mathrm{rk}(\varphi)$ if x is a structure variable, $\mathrm{rk}(\exists x\varphi) = \mathrm{rk}(\forall x\varphi) = \mathrm{rk}(\varphi)$ if x is a number variable, and $\mathrm{rk}(\#\bar{x}\,\varphi = \bar{p}) = |\bar{x}| + \mathrm{rk}(\varphi)$. Now, a $\mathcal{L}_{\infty\omega}(\mathbf{C})[\tau]$-formula φ belongs to $\mathcal{L}^*_{\infty\omega}(\mathbf{C})[\tau]$ if there is a number $n \in \mathbb{N}$ with $\mathrm{rk}(\varphi) \le n$.

As shown in [21], every $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$ formula without free number variables is Gaifman local. To make this precise, we need some more notation. Given a graph G and vertices $v, w \in V(G)$, let $\mathrm{dist}^G(v, w)$ denote the length of a shortest path from v to w in the undirected graph obtained from G by adding edges (w', v') for every edge $(v', w') \in E(G)$, or ∞ if there is no such path. For all $k \ge 1$, all tuples $\bar{v} = (v_1, \ldots, v_k) \in V(G)^k$ and all $r \in \mathbb{N}$, let $B^G_r(\bar{v}) := \{w \in V(G) \mid \exists i \in [k]: \mathrm{dist}^G(v_i, w) \le r\}$, and define $N^G_r(\bar{v})$ to be the subgraph of G induced by $B^G_r(\bar{v})$. The following theorem is stated in [21] for arbitrary vocabularies:

Theorem 5.3 (Restricted form of a theorem in [21]). *For every $\mathcal{L}^*_{\infty\omega}(\boldsymbol{C})[\{E\}]$-formula $\varphi(\bar{x})$ without free number variables, there is an $r \in \mathbb{N}$ such that for all graphs G and all $\bar{a}, \bar{b} \in V(G)^{|\bar{x}|}$ with $(N_r^G(\bar{a}), \bar{a}) \cong (N_r^G(\bar{b}), \bar{b})$ we have: $G \models \varphi[\bar{a}] \iff G \models \varphi[\bar{b}]$.*

Using Theorem 5.3, it is straightforward to show that:

Corollary 5.4. *There is no $\mathcal{L}^*_{\infty\omega}(\boldsymbol{C})[\{E\}]$-formula $\varphi(x, y)$ such that for all $G \in \mathcal{C}$ and all $v, w \in V(G)$ we have $G \models \varphi[v, w]$ iff there is a path from v to w in G.*

Proof. For a contradiction, suppose that $\varphi(x, y)$ is an $\mathcal{L}^*_{\infty\omega}(\boldsymbol{C})[\{E\}]$-formula such that for all $G \in \mathcal{C}$ and all $v, w \in V(G)$ we have $G \models \varphi[v, w]$ iff there is a path from v to w in G. Let $r \in \mathbb{N}$ be as guaranteed by Theorem 5.3. We can now pick vertices $v, w_1, w_2 \in G_{r+2}$ with $N_r^{G_{r+2}}(v, w_1) \cong N_r^{G_{r+2}}(v, w_2)$ such that w_1 is reachable from v, but w_2 is not reachable from v. Since $G_{r+2} \models \varphi[v, w_1]$, we then have $G_{r+2} \models \varphi[v, w_2]$, a contradiction. $\qquad\square$

5.2. Translation of LREC-Formulae Into $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$-Formulae.

We now describe the translation of an LREC-formula φ into an $\mathcal{L}^*_{\infty\omega}(\mathbf{C})$-formula $\tilde{\varphi}$ that is equivalent to φ on $\mathcal{C}$. The translation proceeds by induction on the structure of φ, where the only interesting case is that of LREC-formulae φ of the form

$$[\mathsf{lrec}_{\bar{u}_1, \bar{u}_2, \bar{p}} \; \varphi_{\mathsf{E}}, \; \varphi_{\mathsf{C}}](\bar{w}, \bar{r}).$$

To decide whether φ holds in a given graph G_n under an assignment α, $\tilde{\varphi}$ needs to check whether the tuple $(\bar{a}_0, \ell_0)$, for $\bar{a}_0 := \alpha(\bar{w})$ and $\ell_0 := \langle \alpha(\bar{r}) \rangle$, belongs to the relation X defined by φ in (G_n, α). To this end, it looks at the graph G with vertex set $G_n^{\bar{u}_1}$ and edge set $\varphi_{\mathsf{E}}[G_n, \alpha; \bar{u}_1, \bar{u}_2]$, or rather at its ℓ_0-*unravelling* $\mathsf{G}^{(\bar{a}_0, \ell_0)}$ at $\bar{a}_0$:

Definition 5.5. The ℓ-*unravelling* of a graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ at a vertex $v \in \mathsf{V}$ is the tree $\mathsf{G}^{(v, \ell)}$ defined as follows:

(1) The nodes of $\mathsf{G}^{(v, \ell)}$ are all finite sequences $((v_0, \ell_0), \ldots, (v_n, \ell_n))$, where $(v_0, \ell_0) = (v, \ell)$, $(v_0, \ldots, v_n)$ is a path in G, and $\ell_i = \lfloor (\ell_{i-1} - 1)/|\mathsf{E}v_i| \rfloor$ for every $i \in [n]$.
(2) There is an edge from a node $((v_0, \ell_0), \ldots, (v_m, \ell_m))$ to a node $((v'_0, \ell'_0), \ldots, (v'_n, \ell'_n))$ whenever $n = m + 1$, and $(v'_i, \ell'_i) = (v_i, \ell_i)$ for every $i \leq m$.
(3) Each node $((v_0, \ell_0), \ldots, (v_m, \ell_m))$ is labelled with (v_m, ℓ_m).

For each node of $\mathsf{G}^{(\bar{a}_0, \ell_0)}$, $\tilde{\varphi}$ checks whether its label belongs to X. Clearly, this suffices to decide whether $(\bar{a}_0, \ell_0) \in X$.

Our construction is based on the following property of $\mathsf{G}^{(\bar{a}_0, \ell_0)}$:

Lemma 5.6. *Let $\varphi_{\mathsf{E}}(\bar{x}, \bar{y}, \bar{z})$ be a formula, where $\bar{x}, \bar{y}$ are compatible, let $n > |\bar{x}| + |\bar{z}| + 2$, let α be an assignment for φ_{E} in G_n, and let $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ be the graph with $\mathsf{V} := G_n^{\bar{x}}$ and $\mathsf{E} := \varphi_{\mathsf{E}}[G_n, \alpha; \bar{x}, \bar{y}]$. Consider a node $((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m))$ in $\mathsf{G}^{(\bar{a}, \ell)}$, where $\ell \leq |N(G_n)|^r - 1$. Then, the size of*

$$\mathcal{I} := \{i \in [m] \mid (\tilde{a}_{i-1} \cup \alpha(\tilde{z})) \cap V(G_n) \neq (\tilde{a}_i \cup \alpha(\tilde{z})) \cap V(G_n)\}$$

is bounded by a constant that depends only on φ_{E} and r.

Proof. We first show that the size of

$$\mathcal{K} := \{i \in \mathcal{I} \mid \tilde{a}_{i-1} \cap V(G_n) \nsubseteq (\tilde{a}_i \cup \alpha(\tilde{z})) \cap V(G_n)\}$$

is bounded by a constant that only depends on φ_{E} and r. To this end, consider an $i \in \mathcal{K}$ and a $b \in \tilde{a}_{i-1} \cap V(G_n)$ such that $b \notin \tilde{a}_i \cup \alpha(\tilde{z})$. Let us call an element $b' \in V(G_n)$ a *sibling* of b if b and b' belong to the same layer in G_n. There are at least

$$n - |\tilde{a}_i \cup \alpha(\tilde{z})| - 1 \geq n - (|\bar{x}| + |\bar{z}| + 1)$$

siblings of b in G_n that do not occur in $\tilde{a}_i \cup \alpha(\tilde{z}) \cup \{b\}$. Each such sibling b' gives rise to an *automorphism* $f_{b'} \colon V(G_n) \to V(G_n)$ of G_n that fixes all the vertices in $V(G_n) \setminus \{b, b'\}$ point-wise, maps b to b', and maps b' to b. As a consequence, for each such sibling b' we have $f_{b'}(\bar{a}_{i-1})\bar{a}_i \in \mathsf{E}$, where $f_{b'}(\bar{a}_{i-1})$ is the tuple obtained from $\bar{a}_{i-1}$ by replacing each element b'' in $\bar{a}_{i-1}$ that belongs to $V(G_n)$ with $f_{b'}(b'')$. This implies

$$|\mathsf{E}\bar{a}_i| \geq n - d_1,$$

where $d_1 := |\bar{x}| + |\bar{z}| + 1$ depends only on φ_{E}.

Observe that, by the definition of $\mathsf{G}^{(\bar{a}, \ell)}$, we have $\ell_0 = \ell \leq |N(G_n)|^r - 1 \leq (2n)^{2r}$ and $\ell_0 \geq \prod_{i=1}^m |\mathsf{E}\bar{a}_i|$. Hence,

$$(2n)^{2r} \;\geq\; \prod_{i=1}^m |\mathsf{E}\bar{a}_i| \;\geq\; \prod_{i \in \mathcal{K}} |\mathsf{E}\bar{a}_i| \;\geq\; \prod_{i \in \mathcal{K}} (n - d_1) \;=\; (n - d_1)^{|\mathcal{K}|}.$$

For $n > d_1 + 1$ this implies $|\mathcal{K}| \leq \log_{n-d_1}(2n)^{2r} \leq 2r(1 + \log_{n-d_1} n)$, which is bounded by a constant d_2 that only depends on φ_{E} and r.

To conclude the proof, consider a maximal set $\mathcal{I}' \subseteq \mathcal{I}$ such that there are no $i, i' \in \mathcal{I}'$ and $k \in \mathcal{K}$ with $i \leq k \leq i'$. We show that $|\mathcal{I}'|$ is bounded by a constant d_3 that depends only on φ_{E}. This then implies the lemma as

$$|\mathcal{I}| \leq (|\mathcal{K}| + 1) \cdot (d_3 + 1) \leq (d_2 + 1) \cdot (d_3 + 1).$$

Let $i_{\min} := \min \mathcal{I}'$ and $i_{\max} := \max \mathcal{I}'$, and notice that

$$\big(\tilde{a}_{i_{\min}-1} \cup \alpha(\tilde{z})\big) \cap V(G_n) \;\subseteq\; \big(\tilde{a}_{i_{\min}} \cup \alpha(\tilde{z})\big) \cap V(G_n) \;\subseteq\; \cdots \;\subseteq\; \big(\tilde{a}_{i_{\max}} \cup \alpha(\tilde{z})\big) \cap V(G_n).$$

Since $\big(\tilde{a}_{i_{\max}} \cup \alpha(\tilde{z})\big) \cap V(G_n)$ contains at most $d_3 := |\bar{x}|$ elements that do not belong to $\big(\tilde{a}_{i_{\min}-1} \cup \alpha(\tilde{z})\big) \cap V(G_n)$, there are at most d_3 indices $i \in [i_{\min}, i_{\max}]$ with $\big(\tilde{a}_{i-1} \cup \alpha(\tilde{z})\big) \cap V(G_n) \subsetneqq \big(\tilde{a}_i \cup \alpha(\tilde{z})\big) \cap V(G_n)$. Hence, $|\mathcal{I}'| \leq d_3$, as desired. $\qquad\square$

We are now ready to prove that on $\mathcal{C}$, every $\mathsf{LREC}[\{E\}]$-formula is equivalent to a $\mathcal{L}^*_{\infty\omega}(\mathbf{C})[\{E\}]$-formula.

Lemma 5.7. *For every $\mathsf{LREC}[\{E\}]$-formula $\varphi(\bar{x})$, there is a $\mathcal{L}^*_{\infty\omega}(\mathbf{C})[\{E\}]$-formula $\tilde{\varphi}(\bar{x})$ such that for all $G \in \mathcal{C}$ and all $\bar{a} \in G^{\bar{x}}$, we have: $G \models \varphi[\bar{a}] \iff G \models \tilde{\varphi}[\bar{a}]$.*

Proof. As mentioned above, we proceed by induction on the structure of φ. The only interesting case is that of an $\mathsf{LREC}[\{E\}]$-formula of the form

$$\varphi = [\mathsf{lrec}_{\bar{u}_1, \bar{u}_2, \bar{p}} \; \varphi_{\mathsf{E}}, \; \varphi_{\mathsf{C}}](\bar{w}, \bar{r}).$$

Let $\bar{v}_{\mathsf{E}}$ be an enumeration of all variables in $\mathrm{free}(\varphi_{\mathsf{E}})$ that are not listed in $\bar{u}_1\bar{u}_2$, and let $\bar{v}_{\mathsf{C}}$ be an enumeration of all variables in $\mathrm{free}(\varphi_{\mathsf{C}})$ that are not listed in $\bar{u}_1\bar{p}$.

We aim to construct, for all integers $n \geq 1$ and $\ell \leq |N(G_n)|^{|\bar{r}|} - 1$, a $\mathcal{L}^*_{\infty\omega}(\mathbf{C})[\{E\}]$-formula $\psi_{n,\ell}(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$ such that for all assignments α in G_n, and all $\bar{a} \in G_n^{\bar{u}_1}$,

$$G_n \models \psi_{n,\ell}[\bar{a}, \alpha(\bar{v}_{\mathsf{E}}), \alpha(\bar{v}_{\mathsf{C}})] \iff (\bar{a}, \ell) \in X,$$

where X is the relation defined by φ in (G_n, α). Furthermore, the rank of each $\psi_{n,\ell}$ will be bounded by a constant that depends only on φ, so that

$$\tilde{\varphi} := \bigvee_{\substack{n \geq 1 \\ \ell < (2n^2+1)^{|\bar{r}|}}} \Big(\text{``the universe has size } 2n^2\text{''} \wedge \text{``}\bar{r} \text{ represents the number } \ell\text{''} \wedge \psi_{n,\ell}(\bar{w}, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}}) \Big)$$

is a $\mathcal{L}_{\infty\omega}^*(\mathbf{C})[\{E\}]$-formula that is equivalent to φ on $\mathcal{C}$.

Construction of $\psi_{n,\ell}(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$: Fix $n \geq 1$ and $\ell \leq |N(G_n)|^{|\bar{r}|} - 1$. To simplify the presentation, we also fix an assignment α in G_n, and the graph $\mathtt{G} = (\mathtt{V}, \mathtt{E})$ with $\mathtt{V} := G_n^{\bar{u}_1}$ and $\mathtt{E} := \varphi_{\mathsf{E}}[G_n, \alpha; \bar{u}_1, \bar{u}_2]$; the formula $\psi_{n,\ell}(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$ we are going to construct will however not depend on α. For every $\bar{a} \in \mathtt{V}$, let

$$t_{n,\ell}(\bar{a}) := \max \{ t \in \mathbb{N} \mid \text{there is a node } (\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m) \text{ in } \mathtt{G}^{(\bar{a},\ell)} \text{ such that } t \text{ equals}$$
$$|\{ i \in [m] \mid (\tilde{a}_{i-1} \cup \alpha(\tilde{v}_{\mathsf{E}})) \cap V(G_n) \neq (\tilde{a}_i \cup \alpha(\tilde{v}_{\mathsf{E}})) \cap V(G_n) \}|\}.$$

By Lemma 5.6 there is a constant t^* that only depends on φ such that

$$t_{n,\ell}(\bar{a}) < t^* \quad \text{for all } \bar{a} \in \mathtt{V}.$$

In what follows, we construct, for all $t \leq t^*$, a $\mathcal{L}_{\infty\omega}^*(\mathbf{C})[\{E\}]$-formula $\psi_{n,\ell}^t(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$ such that for all $\bar{a} \in \mathtt{V}$ with $t_{n,\ell}(\bar{a}) < t$, we have:

$$G_n \models \psi_{n,\ell}^t[\bar{a}, \alpha(\bar{v}_{\mathsf{E}}), \alpha(\bar{v}_{\mathsf{C}})] \iff (\bar{a}, \ell) \in X,$$

where X is the relation defined by φ in (G_n, α). Furthermore, the rank of $\psi_{n,\ell}^t$ will not depend on n or ℓ. The desired formula $\psi_{n,\ell}$ can then be defined as:

$$\psi_{n,\ell} := \psi_{n,\ell}^{t^*}.$$

Construction of $\psi_{n,\ell}^t(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$: We construct the formulae $\psi_{n,\ell}^t(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$ by induction on t. For $t = 0$, we define $\psi_{n,\ell}^0(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$ to be an arbitrary unsatisfiable formula. The idea for the construction of $\psi_{n,\ell}^{t+1}(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$ is as follows. Let $\bar{a} \in \mathtt{V}$, and

$$\mathcal{Q}(\bar{a}) := \{ (\bar{a}_m, \ell_m) \mid ((\bar{a}_0, \ell_0), \ldots, (\bar{a}_m, \ell_m)) \in V(\mathtt{G}^{(\bar{a},\ell)}), \text{ and for all } i \in [m] \text{ we have:}$$
$$(\tilde{a}_{i-1} \cup \alpha(\tilde{v}_{\mathsf{E}})) \cap V(G_n) = (\tilde{a}_i \cup \alpha(\tilde{v}_{\mathsf{E}})) \cap V(G_n)\}.$$

To check whether $(\bar{a}, \ell) \in X$, we "guess" the set $\hat{X} = \mathcal{Q}(\bar{a}) \cap X$, and then simply check whether $(\bar{a}, \ell) \in \hat{X}$. To guess $\hat{X}$, we can use an infinite disjunction over all subsets R of $\mathcal{Q}(\bar{a})$. Then we only need to verify for each R whether R indeed corresponds to $\hat{X}$. For the latter, we count, for each pair $(\bar{a}', \ell') \in \mathcal{Q}(\bar{a})$, the number of pairs $(\bar{a}'', \ell'')$ such that $\bar{a}'\bar{a}'' \in \mathtt{E}$, $\ell'' = \lfloor (\ell' - 1)/|\mathtt{E}\bar{a}''| \rfloor$ and $(\bar{a}'', \ell'') \in X$, and check that $(\bar{a}', \ell') \in R$ whenever this number belongs to the label of $\bar{a}'$ defined by φ_{C}. How do we check whether $(\bar{a}'', \ell'') \in X$? If $(\tilde{a}' \cup \alpha(\tilde{v}_{\mathsf{E}})) \cap V(G_n) = (\tilde{a}'' \cup \alpha(\tilde{v}_{\mathsf{E}})) \cap V(G_n)$, that is, if $(\bar{a}'', \ell'') \in \mathcal{Q}(\bar{a})$, then we simply check whether $(\bar{a}'', \ell'') \in R$. Otherwise, we use the formula $\psi_{n,\ell''}^t$.

Let φ_{E}' and φ_{C}' be $\mathcal{L}_{\infty\omega}^*(\mathbf{C})[\{E\}]$-formulae that are equivalent to φ_{E} and φ_{C}, respectively. Such formulae exist by the induction hypothesis. Using φ_{E}' it is easy to construct, for each $\ell' \in [0, \ell]$, an $\mathcal{L}_{\infty\omega}^*(\mathbf{C})[\{E\}]$-formula $\chi_{\ell'}(\bar{u}_1, \bar{u}_1', \bar{v}_{\mathsf{E}})$ such that for all $\bar{a}, \bar{a}' \in G_n^{\bar{u}_1}$,

$$G_n \models \chi_{\ell'}[\bar{a}, \bar{a}', \alpha(\bar{v}_{\mathsf{E}})] \iff (\bar{a}', \ell') \in \mathcal{Q}(\bar{a}).$$

Here, $\bar{u}_1'$ is a tuple of variables that is compatible with, but disjoint from $\bar{u}_1$.

Let $\mathcal{Q}'$ be the set of all pairs $(\bar{u}', \ell')$, where $\ell' \in [0, \ell]$, and $\bar{u}'$ is obtained from $\bar{u}'_1$ by replacing each structure variable with a structure variable from $\bar{u}_1$ and each number variable with an integer from $N(G_n) = [0, 2n^2]$. Intuitively, each $R \subseteq \mathcal{Q}'$ corresponds to a guess of $\mathcal{Q}(\bar{a}) \cap X$ as described above. For each $R \subseteq \mathcal{Q}'$, let $\psi_{n,\ell,R}^{t+1}(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$ be

$$\bigwedge_{(\bar{u}', \ell') \in R} \left(\chi_{\ell'}(\bar{u}_1, \bar{u}', \bar{v}_{\mathsf{E}}) \to \exists \bar{p} \Big(\varphi'_{\mathsf{C}}(\bar{u}', \bar{p}, \bar{v}_{\mathsf{C}}) \wedge \#\bar{u}''(\varphi'_{\mathsf{E}}(\bar{u}', \bar{u}'', \bar{v}_{\mathsf{E}}) \wedge \vartheta_{R, \bar{u}', \ell'}(\bar{u}'')) = \bar{p} \Big) \right)$$

$$\wedge \bigwedge_{\substack{(\bar{u}', \ell') \in \mathcal{Q}' \\ (\bar{u}', \ell') \notin R}} \left(\chi_{\ell'}(\bar{u}_1, \bar{u}', \bar{v}_{\mathsf{E}}) \to \exists \bar{p} \Big(\neg\varphi'_{\mathsf{C}}(\bar{u}', \bar{p}, \bar{v}_{\mathsf{C}}) \wedge \#\bar{u}''(\varphi'_{\mathsf{E}}(\bar{u}', \bar{u}'', \bar{v}_{\mathsf{E}}) \wedge \vartheta_{R, \bar{u}', \ell'}(\bar{u}'')) = \bar{p} \Big) \right)$$

where

$$\vartheta_{R, \bar{u}', \ell'}(\bar{u}'') \; := $$
$$\bigvee_{\ell'' \in [0, \ell']} \left(\text{``}\ell'' = \left\lfloor \frac{\ell' - 1}{|\mathsf{E}\bar{u}''|} \right\rfloor \text{''} \wedge \Big( (\chi_{\ell''}(\bar{u}', \bar{u}'', \bar{v}_{\mathsf{E}}) \wedge \text{``}(\bar{u}'', \ell'') \in R \text{''}) \vee \psi_{n, \ell''}^{t}(\bar{u}'', \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}}) \Big) \right)$$

and "$(\bar{u}'', \ell'') \in R$" stands for $\bigvee_{(\bar{u}^\star, \ell^\star) \in R, \, \ell^\star = \ell''} \bar{u}^\star = \bar{u}''$. Then it is not hard to see that the formula

$$\psi_{n, \ell}^{t+1}(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}}) \; := \bigvee_{\substack{R \subseteq \mathcal{Q}' \\ (\bar{u}_1, \ell) \in R}} \psi_{n, \ell, R}^{t+1}(\bar{u}_1, \bar{v}_{\mathsf{E}}, \bar{v}_{\mathsf{C}})$$

is as desired. Clearly, the rank of $\psi_{n, \ell}^{t+1}$ does not depend on n or ℓ. $\square$

To conclude this section, note that Theorem 5.1 follows immediately from Lemma 5.7 and Corollary 5.4.

6. An Extension of LREC

The proof of the previous section's Theorem 5.1 indicates that LREC is not closed under logical reductions, not even under very simple first-order reductions.[7] Indeed, it is easy to see that there is a first-order reduction that maps a graph G_n, for $n \geq 3$, as defined in Section 5 to a disjoint union $\hat{G}_n$ of two directed paths on n vertices each, by identifying vertices in the same layer. Reachability on the class of all graphs isomorphic to $\hat{G}_n$ for an $n \geq 3$ is easily seen to be LREC-definable. Hence, if LREC was closed under first-order reductions, then reachability on the class of all graphs isomorphic to G_n for some n would be LREC-definable, contradicting the previous section's results.

In this section, we introduce an extension LREC$_=$ of LREC whose data complexity is still in LOGSPACE, and thus captures LOGSPACE on directed trees, while being closed under logical reductions. The idea is to admit a third formula $\varphi_=$ in the lrec-operator that generates an equivalence relation on the vertices of the graph defined by φ_{E}.

Let τ be a vocabulary. The set of all LREC$_=[\tau]$-formulae is obtained from LREC$[\tau]$ by replacing the rule for the lrec-operator from Section 3 as follows: If $\bar{u}, \bar{v}, \bar{w}$ are compatible

---

[7]We defer the definition of logical reductions and what it means to be closed under logical reductions to Definition 6.4 and Lemma 6.6. For first-order reduction, see also [5].

tuples of variables, $\bar{p}, \bar{r}$ are non-empty tuples of number variables, and $\varphi_=$, $\varphi_{\mathtt{E}}$ and $\varphi_{\mathtt{C}}$ are LREC$_=$-formulae, then the following is an LREC$_=[\tau]$-formula:

$$\varphi := [\mathsf{lrec}_{\bar{u}, \bar{v}, \bar{p}} \; \varphi_=, \; \varphi_{\mathtt{E}}, \; \varphi_{\mathtt{C}}](\bar{w}, \bar{r}). \tag{6.1}$$

We let $\mathrm{free}(\varphi) := \big(\mathrm{free}(\varphi_=) \setminus (\tilde{u} \cup \tilde{v})\big) \cup \big(\mathrm{free}(\varphi_{\mathtt{E}}) \setminus (\tilde{u} \cup \tilde{v})\big) \cup \big(\mathrm{free}(\varphi_{\mathtt{C}}) \setminus (\tilde{u} \cup \tilde{p})\big) \cup \tilde{w} \cup \tilde{r}$.

To define the semantics of LREC$_=[\tau]$-formulae φ of the form (6.1), let A be a τ-structure and α an assignment in A. Let $\mathtt{V}_0 := A^{\bar{u}}$ and $\mathtt{E}_0 := \varphi_{\mathtt{E}}[A, \alpha; \bar{u}, \bar{v}]$. We define $\sim$ to be the reflexive, symmetric, transitive closure of the binary relation $\varphi_=[A, \alpha; \bar{u}, \bar{v}]$ over $\mathtt{V}_0$. Now consider the graph $\mathtt{G} = (\mathtt{V}, \mathtt{E})$ with

$$\mathtt{V} := \mathtt{V}_0/_\sim \quad \text{and} \quad \mathtt{E} := \{(\bar{a}/_\sim, \bar{b}/_\sim) \in \mathtt{V}^2 \mid \bar{a}\bar{b} \in \mathtt{E}_0\}.$$

To every $\bar{a}/_\sim \in \mathtt{V}$ we assign the set

$$\mathtt{C}(\bar{a}/_\sim) := \{\langle \bar{n} \rangle \mid \text{there is an } \bar{a}' \in \bar{a}/_\sim \text{ with } \bar{n} \in \varphi_{\mathtt{C}}[A, \alpha[\bar{a}'/\bar{u}]; \bar{p}]\}$$

of labels. Then the definition of X can be taken verbatim from Section 3. We let $(A, \alpha) \models \varphi$ if and only if $\big(\alpha(\bar{w})/_\sim, \langle \alpha(\bar{r}) \rangle\big) \in X$. As for LREC, we have:

Theorem 6.1. *For every vocabulary τ, and every LREC$_=[\tau]$-formula φ there is a deterministic logspace Turing machine that, given a τ-structure A and an assignment α in A, decides whether $(A, \alpha) \models \varphi$.*

Sketch. The proof is a straightforward modification of the proof of Theorem 3.4. The only difference is that, when we deal with LREC$_=$-formulae of form (6.1), we use the vertex set $\mathtt{V}$, the edge set $\mathtt{E}$, and the labels $\mathtt{C}(\cdot)$ as defined above to compute the set X. It is easy to compute these sets by first computing the relation $\sim$ from $\varphi_=[A, \alpha; \bar{u}, \bar{v}]$ using Reingold's logspace algorithm for undirected reachability [25]. Note that once $\sim$ has been obtained, the equivalence class of every element $\bar{a} \in A^{\bar{u}}$ can be determined. $\qquad\square$

The following example shows that undirected graph reachability is definable in LREC$_=$. This does not involve an implementation of Reingold's algorithm in our logic, but just uses the observation that the computation of the equivalence relation $\sim$ boils down to the computation of undirected reachability.

Example 6.2 (Undirected reachability)**.** The following LREC$_=$-formula defines undirected graph reachability:

$$\varphi(s, t) := [\mathsf{lrec}_{x,y,p} \; \varphi_=(x, y), \; \varphi_{\mathtt{E}}(x, y), \; \varphi_{\mathtt{C}}(x, p)](s, 1),$$

where $\varphi_=(x, y) := E(x, y)$, $\varphi_{\mathtt{E}}(x, y) := \neg x = x$ and $\varphi_{\mathtt{C}}(x, p) := x = t$. To see this, let G be an undirected graph and α an assignment in G. Define $\sim$, $\mathtt{V}$, $\mathtt{E}$, $\mathtt{C}$ and the set X as above. Clearly, the set $\mathtt{V}$ consists of the connected components of G. Furthermore, the set $\mathtt{E}$ is empty since φ_E is unsatisfiable. Therefore, for all $v \in V(G)$ we have $(v/_\sim, 1) \in X$ iff $0 \in \mathtt{C}(v/_\sim)$. The latter is true precisely if $\alpha(t) \in v/_\sim$, i.e., if v and $\alpha(t)$ are in the same connected component of G. It follows that for all $v, w \in V(G)$ we have $G \models \varphi[v, w]$ if and only if v and w are in the same connected component of G, that is, if there is a path from v to w in G. $\qquad\square$

Remark 6.3. It follows immediately from the previous example that STC+C $\leq$ LREC$_=$. Actually, the containment is strict, because LREC $\not\leq$ STC+C by Corollary 4.6. Since in STC+C (and actually in STC) it is possible to transform trees into directed trees, the results from Section 4 imply that LREC$_=$ captures LOGSPACE on the class of all trees, directed as well as undirected. Note also that LREC$_=$ $\leq$ FP+C.

To conclude this section, we show that $\mathsf{LREC}_=$ is closed under logical reductions. We first introduce L-*transductions* (also known as L-*interpretations* [5]):

Definition 6.4 (Transduction). Let L be a logic, let τ_1, τ_2 be vocabularies and let $\ell \geq 1$.
(1) An ℓ-ary $\mathsf{L}[\tau_1, \tau_2]$-*transduction* is a tuple

$$\Theta = \Big(\theta_V(\bar{u}), \theta_\approx(\bar{u}, \bar{v}), \big(\theta_R(\bar{u}_{R,1}, \ldots, \bar{u}_{R,\mathrm{ar}(R)})\big)_{R \in \tau_2}\Big)$$

of $\mathsf{L}[\tau_1]$-formulae, where $\bar{u}, \bar{v}$ are compatible ℓ-tuples of variables and for every $R \in \tau_2$ and $i \in [\mathrm{ar}(R)]$, $\bar{u}_{R,i}$ is an ℓ-tuple of variables that is compatible to $\bar{u}$.
(2) Let A be a τ_1-structure such that $\theta_V[A; \bar{u}]$ is non-empty. We define a τ_2-structure $\Theta[A]$ as follows. We let $\approx$ be the reflexive, symmetric, transitive closure of the binary relation $\theta_\approx[A; \bar{u}, \bar{v}]$, and call $\approx$ the equivalence relation *generated* by $\theta_\approx[A; \bar{u}, \bar{v}]$. Let

$$V(\Theta[A]) := \theta_V[A; \bar{u}]/_\approx,$$

and for each $R \in \tau_2$, let

$$R(\Theta[A]) := \{(\bar{a}_1/_\approx, \ldots, \bar{a}_{\mathrm{ar}(R)}/_\approx) \mid$$
$$\bar{a}_1, \ldots, \bar{a}_{\mathrm{ar}(R)} \in \theta_V[A; \bar{u}], \ A \models \theta_R[\bar{a}_1, \ldots, \bar{a}_{\mathrm{ar}(R)}]\}.$$

So, informally, a $\mathsf{L}[\tau_1, \tau_2]$-transduction defines a mapping from structures over the first vocabulary, τ_1, into structures over the second vocabulary, τ_2, via $\mathsf{L}[\tau_1]$-formulae.

Example 6.5. Consider the $\mathsf{FO}[\{E\}, \{E\}]$-transduction $\Theta = (\theta_V(x), \theta_\approx(x, y), \theta_E(x, y))$ with $\theta_V(x) := x = x$, $\theta_\approx(x, y) := \forall z\big(E(x, z) \leftrightarrow E(y, z)\big)$ and $\theta_E(x, y) := E(x, y)$. Recall the definition of the graphs G_n from Section 5. For $n > 3$, the equivalence relation $\approx$ generated by $\theta_\approx[G_n; x, y]$ is $\theta_\approx[G_n; x, y]$ itself. It relates any two vertices that occur in the same layer of G_n. Hence, for $n > 3$, $\Theta[G_n]$ is the disjoint union of two paths of length n. □

The following lemma shows that $\mathsf{LREC}_=$ is closed under $\mathsf{LREC}_=$-reductions. Precisely, this means that:

Lemma 6.6. *Let τ_1, τ_2 be vocabularies, let $\ell \geq 1$, let*

$$\Theta = \Big(\theta_V(\bar{u}), \theta_\approx(\bar{u}, \bar{v}), \big(\theta_R(\bar{u}_{R,1}, \ldots, \bar{u}_{R,\mathrm{ar}(R)})\big)_{R \in \tau_2}\Big)$$

be an ℓ-ary $\mathsf{LREC}_=[\tau_1, \tau_2]$-transduction, and let $\varphi(x_1, \ldots, x_\kappa, p_1, \ldots, p_\lambda)$ be an $\mathsf{LREC}_=[\tau_2]$-formula with $x_1, \ldots, x_\kappa$ structure variables and $p_1, \ldots, p_\lambda$ number variables.

Then there is an $\mathsf{LREC}_=[\tau_1]$-formula $\varphi^{-\Theta}(\bar{u}_1, \ldots, \bar{u}_\kappa, \bar{q}_1, \ldots, \bar{q}_\lambda)$, where $\bar{u}_1, \ldots, \bar{u}_\kappa$ are compatible with $\bar{u}$ and $\bar{q}_1, \ldots, \bar{q}_\lambda$ are ℓ-tuples of number variables, such that for all τ_1-structures A where $\Theta[A]$ is defined, all $\bar{a}_1, \ldots, \bar{a}_\kappa \in A^{\bar{u}}$ and all $\bar{n}_1, \ldots, \bar{n}_\lambda \in N(A)^\ell$,

$$A \models \varphi^{-\Theta}[\bar{a}_1, \ldots, \bar{a}_\kappa, \bar{n}_1, \ldots, \bar{n}_\lambda] \iff \bar{a}_1/_\approx, \ldots, \bar{a}_\kappa/_\approx \in V(\Theta[A]),$$
$$\langle \bar{n}_1 \rangle_A, \ldots, \langle \bar{n}_\lambda \rangle_A \in N(\Theta[A]), \ and$$
$$\Theta[A] \models \varphi[\bar{a}_1/_\approx, \ldots, \bar{a}_\kappa/_\approx, \langle \bar{n}_1 \rangle_A, \ldots, \langle \bar{n}_\lambda \rangle_A],$$

where $\approx$ is the equivalence relation as defined in Definition 6.4.

Proof. The proof is by induction on the structure of φ. Without loss of generality, we assume that φ neither contains implication ($\rightarrow$) nor biimplication ($\leftrightarrow$).

To simplify the presentation, we consider a fixed τ_1-structure A where $\Theta[A]$ is defined and let $\approx$ be the equivalence relation as defined in Definition 6.4. We also consider fixed

$\bar{a}_1, \ldots, \bar{a}_\kappa \in A^{\bar{u}}$ and $\bar{n}_1, \ldots, \bar{n}_\lambda \in N(A)^\ell$. The reader should consider A and these tuples to be universally quantified in the statements where they occur.

From $\theta_\approx(\bar{u}, \bar{v})$, it is easy to construct an $\mathsf{LREC}_=$-formula $\theta'_\approx(\bar{u}, \bar{v})$ such that $\theta'_\approx[A; \bar{u}, \bar{v}]$ is the equivalence relation generated by $\theta_\approx[A; \bar{u}, \bar{v}]$, that is, the reflexive, symmetric, and transitive closure of $\theta_\approx(\bar{u}, \bar{v})$. Let $\chi_s(\bar{u}) := \exists \bar{v} \left(\theta_V(\bar{v}) \wedge \theta'_\approx(\bar{u}, \bar{v})\right)$. Then for all $\bar{a} \in A^{\bar{u}}$,

$$A \models \chi_s[\bar{a}] \iff \bar{a}/_\approx \in V(\Theta[A]).$$

Using the construction from the proof of [20, Lemma 2.4.3], we can construct an $\mathsf{LREC}_=$-formula $\delta_V^\#(\bar{q})$ such that for all $\bar{n} \in N(A)^\ell$ we have $A \models \delta_V^\#[\bar{n}]$ whenever $\langle \bar{n} \rangle_A = |V(\Theta[A])|$. Hence, for $\chi_n(\bar{q}) := \exists \bar{q}'(\delta_V^\#(\bar{q}') \wedge \text{``}\bar{q} \leq \bar{q}'\text{''})$ and for all $\bar{n} \in N(A)^\ell$,

$$A \models \chi_n[\bar{n}] \iff \langle \bar{n} \rangle_A \in N(\Theta[A]).$$

Finally, let

$$\chi := \bigwedge_{i \in [\kappa]} \chi_s(\bar{u}_i) \wedge \bigwedge_{i \in [\lambda]} \chi_n(\bar{q}_i)$$

Then,

$$A \models \chi[\bar{a}_1, \ldots, \bar{a}_\kappa, \bar{n}_1, \ldots, \bar{n}_\lambda] \iff$$
$$\bar{a}_1/_\approx, \ldots, \bar{a}_\kappa/_\approx \in V(\Theta[A]) \quad \text{and} \quad \langle \bar{n}_1 \rangle_A, \ldots, \langle \bar{n}_\lambda \rangle_A \in N(\Theta[A]).$$

Given $\varphi(x_1, \ldots, x_\kappa, p_1, \ldots, p_\lambda)$ we now construct $\varphi^{-\Theta}(\bar{u}_1, \ldots, \bar{u}_\kappa, \bar{q}_1, \ldots, \bar{q}_\lambda)$ inductively as follows:

(1) Suppose that $\varphi = R(x_{i_1}, \ldots, x_{i_k})$, where $i_1, \ldots, i_k \in [\kappa]$. Let $\mathcal{I} := \{i_1, \ldots, i_k\}$. Then,

$$\varphi^{-\Theta} := \chi \wedge (\exists \bar{v}_i)_{i \in \mathcal{I}} \left(\bigwedge_{i \in \mathcal{I}} \theta'_\approx(\bar{u}_i, \bar{v}_i) \wedge \bigwedge_{i \in \mathcal{I}} \theta_V(\bar{v}_i) \wedge \theta_R(\bar{v}_{i_1}, \ldots, \bar{v}_{i_k}) \right).$$

(2) If $\varphi = x_i = x_j$, where $i, j \in [\kappa]$, then $\varphi^{-\Theta} := \chi \wedge \theta'_\approx(\bar{u}_i, \bar{u}_j)$.

(3) If $\varphi = p_i \star p_j$, where $\star \in \{=, \leq\}$ and $i, j \in [\lambda]$, then, $\varphi^{-\Theta} := \chi \wedge \text{``}\bar{q}_i \star \bar{q}_j\text{''}$.

(4) If $\varphi = \neg \psi$, then $\varphi^{-\Theta} := \chi \wedge \neg \psi^{-\Theta}$.

(5) If $\varphi = \psi_1 \star \psi_2$, where $\star \in \{\wedge, \vee\}$, then $\varphi^{-\Theta} := \psi_1^{-\Theta} \star \psi_2^{-\Theta}$.

(6) Suppose that $\varphi = Qu\,\psi$ with $Q \in \{\forall, \exists\}$ and $u \in \{x_1, \ldots, x_\kappa, p_1, \ldots, p_\lambda\}$. In case that $Q = \forall$ and $u = x_i$, we let $\varphi^{-\Theta} := \chi \wedge \forall \bar{u}_i(\chi_s(\bar{u}_i) \rightarrow \psi^{-\Theta})$. The other cases can be dealt with similarly.

(7) Suppose that $\varphi = \#(x_{i_1}, \ldots, x_{i_k}, p_{i_{k+1}}, \ldots, p_{i_{k+m}})\,\psi = (p_{j_1}, \ldots, p_{j_{k'}})$. Based on the construction from the proof of [20, Lemma 2.4.3], it is possible to construct an $\mathsf{LREC}_=$-formula $\delta(\bar{r}_1, \ldots, \bar{r}_{k'})$ such that for all $\bar{m}_1, \ldots, \bar{m}_{k'} \in N(A)^\ell$,

$$A \models \delta[\bar{m}_1, \ldots, \bar{m}_{k'}]$$
$$\iff \left| \left\{ (\bar{a}_{i_1}/_\approx, \ldots, \bar{a}_{i_k}/_\approx, \langle \bar{n}_{i_{k+1}} \rangle_A, \ldots, \langle \bar{n}_{i_{k+m}} \rangle_A) \mid \right. \right.$$
$$\left. \left. A \models \psi^{-\Theta}[\bar{a}_1, \ldots, \bar{a}_\kappa, \bar{n}_1, \ldots, \bar{n}_\lambda] \right\} \right| = \langle \bar{m}_1, \ldots, \bar{m}_{k'} \rangle_A,$$

where

$$\langle \bar{m}_1, \ldots, \bar{m}_{k'} \rangle_A := \sum_{s=1}^{k'} \langle \bar{m}_s \rangle_A \cdot |N(\Theta[A])|^{s-1}.$$

We then let $\varphi^{-\Theta} := \chi \wedge \delta(\bar{q}_{j_1}, \ldots, \bar{q}_{j_{k'}})$.

(8) Suppose that $\varphi = [\mathsf{lrec}_{\bar{u}',\bar{v}',\bar{p}'} \; \varphi_=, \; \varphi_{\mathsf{E}}, \; \varphi_{\mathsf{C}}](\bar{w}', \bar{r}')$. Then,

$$\varphi^{-\Theta} := \chi \wedge \exists \bar{r}\big(\beta(\bar{r}'', \bar{r}) \wedge [\mathsf{lrec}_{\bar{u}'',\bar{v}'',\bar{p}''} \; \varphi_=^{-\Theta}, \; \varphi_{\mathsf{E}}^{-\Theta}, \; \varphi_{\mathsf{C}}^{-\Theta}](\bar{w}'', \bar{r})\big),$$

where $\bar{u}'', \bar{v}'', \bar{w}'', \bar{p}'', \bar{r}''$ are obtained from $\bar{u}', \bar{v}', \bar{w}', \bar{p}', \bar{r}'$ by replacing, for each $i \in [\kappa]$, the variable x_i by $\bar{u}_i$, and for each $i \in [\lambda]$, the variable p_i by $\bar{q}_i$; $\bar{r}$ is a tuple of number variables of length $\ell \cdot |\bar{r}'|$; and β is defined as follows. For simplicity, assume that $\bar{r}' = (p_1, \ldots, p_k)$. Hence, $\bar{r}'' = (\bar{q}_1, \ldots, \bar{q}_k)$. The formula $\beta(\bar{r}'', \bar{r})$ has the property that for all $\bar{m} \in N(A)^{\ell \cdot k}$,

$$A \models \beta[\bar{n}_1, \ldots, \bar{n}_k, \bar{m}] \iff \langle \bar{m} \rangle_A = \sum_{s=1}^{k} \langle \bar{n}_s \rangle_A \cdot |N(\Theta[A])|^{s-1}.$$

Note that, since $|N(\Theta[A])| \leq |N(A)|^{\ell}$, the tuple $\bar{m}$ is long enough to hold the sum on the right hand side. Constructing β as desired is a not too difficult exercise.

It is straightforward, though tedious, to verify that $\varphi^{-\Theta}$ is as desired. $\qquad\square$

7. Capturing Logspace on Interval Graphs

With the added expressive power of $\mathsf{LREC}_=$, it is not only possible to capture $\mathsf{LOGSPACE}$ on the class of all trees, but also on the class of all interval graphs, as we shall show in this section. Basically, interval graphs are graphs whose vertices are closed intervals, and whose edges join any two distinct intervals with a non-empty intersection. They form a well-established and widely investigated class of graphs, and it was recently shown [18] (see also [20]) that interval graph canonisation is in $\mathsf{LOGSPACE}$.

To prove that $\mathsf{LREC}_=$ captures $\mathsf{LOGSPACE}$ on interval graphs, we proceed as in the case of directed trees. First, we describe an $\mathsf{LREC}_=$-definable canonisation procedure for interval graphs, and then we use the fact that DTC (and hence $\mathsf{LREC}_=$) captures $\mathsf{LOGSPACE}$ on ordered structures. Our canonisation procedure combines algorithmic techniques from [18] with the logical definability framework in [19]. Parts of this section can be found in more detail in [20].
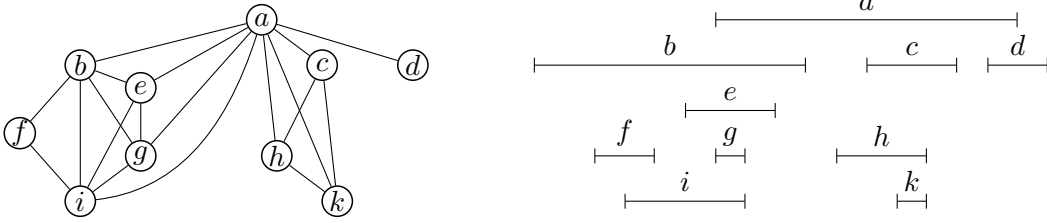
7.1. **Background on Interval Graphs.** In this section, we define interval graphs and state some basic properties. For a more detailed exposition, we refer the reader to [20].

Definition 7.1 (Interval graph, interval representation)**.** Given a finite collection $\mathcal{I}$ of closed intervals $I_i := [a_i, b_i] \subset \mathbb{N}$, let $G_{\mathcal{I}} = (V, E)$ be the graph with vertex set $V = \mathcal{I}$, joining two distinct intervals $I_i, I_j \in V$ by an edge whenever $I_i \cap I_j \neq \emptyset$. We call $\mathcal{I}$ an *interval representation* of a graph G if $G \cong G_{\mathcal{I}}$. A graph G is an *interval graph* if there is an interval representation of G.

Figure 6 shows an interval graph G together with an interval representation of G.

An interval representation $\mathcal{I}$ of a graph G is called *minimal* if the set $\bigcup \mathcal{I} \subset \mathbb{N}$ is of minimum size among all interval representations of G. Clearly, for any interval representation $\mathcal{I}$ there exists a minimal interval representation $\mathcal{I}_{\min}$ such that $G_{\mathcal{I}} \cong G_{\mathcal{I}_{\min}}$.

Recall that a *clique* of a graph $G = (V, E)$ is a set $C \subseteq V$ such that the subgraph of G induced by C is complete. A maximal clique, or *max clique*, of G is a clique of G that is not properly contained in another clique of G. We denote the set of all max cliques of G by $\mathcal{M}_G$. Let $\mathcal{I}$ be a minimal interval representation of G and I_v denote the interval in $\mathcal{I}$ that

Figure 6: An interval graph G and an interval representation of G.

corresponds to vertex $v \in V$. Then $M(k) = \{v \mid k \in I_v\}$ is a max clique of G for every k for which $M(k)$ is non-empty. Furthermore, for any max clique M of G, there is a $k \in \mathbb{N}$ with $M = M(k)$. Thus, any minimal interval representation of G induces a linear order on $\mathcal{M}_G$ which has the property that each vertex is contained in *consecutive* max cliques. It is known [9, 24] that a graph G is an interval graph if and only if its max cliques can be brought into a linear order, so that each vertex of G is contained in consecutive max cliques.

Thus, max cliques play an important role for the structure of interval graphs. Our canonisation procedure essentially relies on bringing the max cliques of an interval graph into a suitable order.

The maximal cliques of an interval graph $G = (V, E)$ can be handled rather easily in our logic. Let $\mathrm{N}^c(v)$ denote the closed neighbourhood of a vertex v in G, that is, the set containing v and all vertices adjacent to v. As shown in [19], the max cliques of G can be identified by the vertex pairs $(u, v) \in V^2$ with the property that $\mathrm{N}^c(u) \cap \mathrm{N}^c(v)$ is a clique in G, and for no other pair $(u', v') \in V^2$ where $\mathrm{N}^c(u') \cap \mathrm{N}^c(v')$ is a clique in G it holds that $\mathrm{N}^c(u) \cap \mathrm{N}^c(v) \subsetneq \mathrm{N}^c(u') \cap \mathrm{N}^c(v')$:

Lemma 7.2 ([19], Lemma IV.1)**.** *Let G be an interval graph and let M be a max clique of G. Then there are vertices $u, v \in M$, not necessarily distinct, such that $M = \mathrm{N}^c(u) \cap \mathrm{N}^c(v)$.*

In particular, the max cliques of G as well as the equivalence relation on vertex pairs defining the same max clique are first-order definable.

7.2. **Modular Decompositions.** Our canonisation procedure relies on a specific decomposition of graphs, known as *modular decomposition*, which was first introduced by Gallai [8]. The basic building blocks of modular decompositions are *modules*. Given a graph $G = (V, E)$, a set $W \subseteq V$ is a *module* of G if for all vertices $v \in V \setminus W$ either $\{v\} \times W \subseteq E$ or $(\{v\} \times W) \cap E = \emptyset$. Note that V and all singleton vertex sets are modules of G, called *trivial modules*. We call a module W proper if $W \subsetneq V$.

Gallai's modular decomposition is based on the following: If G is not connected, then its connected components $W_1, \ldots, W_k$ are clearly proper modules. Similarly, if the complement graph G^c of G is not connected, then the connected components $W_1, \ldots, W_k$ of G^c are proper modules of G. For graphs G with more than one vertex where both G and G^c are connected, Gallai shows in [8] that the set of maximal proper modules of G is a partition of G's vertex set. We base our modular decomposition on the same properties, only for the last one we use a slightly different partition into modules $W_1, \ldots, W_k$, which we define in Section 7.4.[8] Let $\mathcal{W}_G$ be the set of modules $W_1, \ldots, W_k$ and let $\sim_G$ be the equivalence relation on V

---

[8]The main difference between our decomposition and Gallai's is that we do not bother to create extra modules for sets of pairwise connected twins since we can handle them perfectly well with our methods.

corresponding to the partition $\mathcal{W}_G$ (i.e., $v \sim_G w$ whenever $v, w \in W_i$ for some $i \in [k]$). Let us consider the graph

$$L_G := (V/_{\sim_G}, E_{L_G}), \quad \text{where} \quad E_{L_G} := \{(u/_{\sim_G}, v/_{\sim_G}) \mid (u, v) \in E\}.$$

Intuitively, L_G is the graph obtained from G by collapsing all the modules in $\mathcal{W}_G$ into single vertices. Since each pair of modules $W_i, W_j \in \mathcal{W}_G$, $i \neq j$, is either completely connected or completely disconnected, G is completely determined by L_G and the graphs $G[W_i]$, for $i \in [k]$, where $G[W_i]$ denotes the subgraph of G induced by the vertices in W_i. By decomposing the $G[W_i]$, $i \in [k]$, inductively until we arrive at singleton sets everywhere, we obtain G's modular decomposition.

We define the *modular decomposition tree* $T(G)$ of a graph G recursively. If $|V| = 1$, then $T(G)$ is the rooted tree that consists of only one vertex, vertex V, which is the root of $T(G)$. Let $|V| > 1$. Then, the modular decomposition tree $T(G)$ is a rooted tree which consists of a vertex V, which is the root of $T(G)$, and of subtrees $T(G[W])$ for all $W \in \mathcal{W}_G$. We obtain $T(G)$ by adding an edge from V to the root of $T(G[W])$ for all $W \in \mathcal{W}_G$. This modular decomposition tree is uniquely determined for every graph G [8].

Notice that for an interval graph G where G^c is not connected, all except one connected component of G^c must contain only a single vertex. Each of these single vertices is adjacent to all other vertices in G. We call a vertex with that property an *apex*. Thus, if G is an interval graph with G^c disconnected, then $\mathcal{W}_G = \bigcup_{a \in A}\{\{a\}\} \cup \{V \setminus A\}$ where A is the set of apices, and the graph L_G is isomorphic to a clique. Also, if G contains an apex, then either $|V| = 1$ or G^c is not connected.

The following three sections are about defining and canonising the graph L_G for an interval graph G. This is easy for unconnected graphs G or graphs that have at least one apex. Thus, we will consider connected graphs without any apices.

7.3. **Extracting Information About the Order of Maximal Cliques.** Throughout this section let G be a connected interval graph without any apices.

We call a max clique C a possible *end* of G if there is a minimal interval representation $\mathcal{I}$ of G so that C is minimal with respect to the order induced by $\mathcal{I}$.

Now we pick a max clique M of G. We assume it to be a possible end of G, and give a recursive procedure that turns out to recover all the information about the order of the max cliques induced by choosing M as an end of G.

Let $M \in \mathcal{M}_G$. The binary relation $\prec_M$ is defined recursively on the elements of $\mathcal{M}_G$ as follows:

Initialisation: $M \prec_M C$ for all $C \in \mathcal{M}_G \setminus \{M\}$

$$C \prec_M D \quad \text{if} \quad \begin{cases} \exists E \in \mathcal{M}_G \text{ with } E \prec_M D \text{ and } (E \cap C) \setminus D \neq \emptyset \quad \text{or} \\ \exists E \in \mathcal{M}_G \text{ with } C \prec_M E \text{ and } (E \cap D) \setminus C \neq \emptyset. \end{cases} \quad (\bigstar)$$

By exploiting the definition's symmetry, $\prec_M$ can be defined through a reachability query in the undirected graph O_M, which has pairs of max cliques from $\mathcal{M}_G$ as its vertices, and in which two vertices (A, B) and (C, D) are connected by an edge whenever $A \prec_M B$ implies $C \prec_M D$ with one application of $(\bigstar)$. Hence:

Lemma 7.3. *There exists an* STC-*formula that for any interval graph G and for any max clique M of G defines the relation $\prec_M$.*

We now state a few important properties of $\prec_M$. Recall that a binary relation R on a set A is *asymmetric* if $ab \in R$ implies $ba \notin R$ for all $a, b \in A$. In particular, asymmetric relations are irreflexive.

Lemma 7.4 ([19], Lemma IV.3, Corollary IV.6, Lemma IV.7). *Let M be a max clique of an interval graph G. Then the following properties are equivalent:*

- $\prec_M$ *is asymmetric,*
- $\prec_M$ *is a strict weak order (that is, $\prec_M$ is irreflexive, transitive, and incomparability is an equivalence relation),*
- M *is a possible end of G.*

Since $\prec_M$ is STC-definable and asymmetry of $\prec_M$ is FO-definable, the preceding lemma gives us a way to define possible ends of interval graphs in $\mathsf{STC+C}$.

Lemma 7.5. *Let $\mathcal{C} \subset \mathcal{M}_G$ be a set of max cliques with $M \notin \mathcal{C}$. Suppose that for all $B \in \mathcal{M}_G \setminus \mathcal{C}$ and any $C, C' \in \mathcal{C}$ it holds that $B \cap C = B \cap C'$. Then the max cliques in $\mathcal{C}$ are mutually incomparable with respect to $\prec_M$.*

Proof. By a derivation chain of length k we mean a finite sequence $X_0 \prec_M Y_0$, $X_1 \prec_M Y_1$, ..., $X_k \prec_M Y_k$ such that $X_0 = M$ and for each $i \in [k]$, the relation $X_i \prec_M Y_i$ follows from $X_{i-1} \prec_M Y_{i-1}$ by one application of ($\bigstar$). Clearly, whenever it holds that $X \prec_M Y$ there is a derivation chain that has $X \prec_M Y$ as its last element.

Suppose for contradiction that there are $C, C' \in \mathcal{C}$ with $C \prec_M C'$. Let $M \prec_M Y_0$, $X_1 \prec_M Y_1$, ..., $X_k \prec_M Y_k$ be a derivation chain for $C \prec_M C'$. Since $X_k = C$, $Y_k = C'$, and $M \notin \mathcal{C}$, there is a largest index i so that either X_i or Y_i is not contained in $\mathcal{C}$.

If $X_i \notin \mathcal{C}$, then $X_{i+1} \in \mathcal{C}$ and $Y_i = Y_{i+1} \in \mathcal{C}$ and it holds that $X_i \cap X_{i+1} \setminus Y_{i+1} \neq \emptyset$. Consequently, $X_i \cap X_{i+1} \neq X_i \cap Y_{i+1}$, contradicting the assumption of the lemma. Similarly, if $Y_i \notin \mathcal{C}$, then $Y_{i+1} \in \mathcal{C}$ and $X_i = X_{i+1} \in \mathcal{C}$ and it holds that $Y_i \cap Y_{i+1} \setminus X_{i+1} \neq \emptyset$. Thus, $Y_i \cap Y_{i+1} \neq Y_i \cap X_{i+1}$, again a contradiction. $\qquad \square$

The *span* of a vertex $v \in V$ in G, denoted $\mathrm{span}(v)$, is the number of max cliques of G that v is contained in. Recall from Section 7.1 that the equivalence relation on vertex pairs defining the same max clique is first-order definable. Note that, since equivalence classes can be counted in $\mathsf{STC+C}$ [19, Lemma II.7], the span of a vertex is $\mathsf{STC+C}$-definable on the class of all interval graphs.

Lemma 7.6 ([19], Lemma IV.4, Corollary IV.5). *Suppose M is a possible end of G and $\mathcal{C}$ is a maximal set of $\prec_M$-incomparable max cliques. Then*

- $B \cap C = B \cap C'$ *for all $C, C' \in \mathcal{C}$, $B \in \mathcal{M}_G \setminus \mathcal{C}$,*
- $S_{\mathcal{C}} := \bigcup_{C \in \mathcal{C}} C \setminus \bigcup_{B \in \mathcal{M}_G \setminus \mathcal{C}} B$ *is a module of G, and*
- $S_{\mathcal{C}} = \{ v \in \bigcup \mathcal{C} \mid \mathrm{span}(v) \leq |\mathcal{C}| \}$. $\qquad \square$

Finally, let $\sim_M^G$ be the equivalence relation on V for which $x \sim_M^G y$ if and only if $x = y$, or there is a maximal set $\mathcal{C}$ of incomparable max cliques with respect to $\prec_M$ with $|\mathcal{C}| > 1$ so that $x, y \in S_{\mathcal{C}}$. Let $G_M = G/_{\sim_M^G} := (V/_{\sim_M^G}, E_M)$, where $E_M := \{ (u/_{\sim_M^G}, v/_{\sim_M^G}) \mid (u, v) \in E \}$. It is easy to check that $\sim_M^G$ and the graph G_M are $\mathsf{STC+C}$-definable.

If $\mathcal{C}$ is a maximal set of $\prec_M$-incomparables in G with $|\mathcal{C}| > 1$, then there is precisely one max clique $M_{\mathcal{C}}$ in G_M which contains all the equivalence classes associated with $\mathcal{C}$, i.e., $M_{\mathcal{C}} = \{ v/_{\sim_M^G} \mid v \in \bigcup \mathcal{C} \}$. We conclude:

Lemma 7.7. $\prec_M$ *induces a linear order on G_M's max cliques. In particular, G_M is an interval graph.* $\square$

7.4. **Modules $\mathcal{W}_G$ and the Graph L_G.** We are now ready to give the definition of the set $\mathcal{W}_G$, which we mentioned in Section 7.2, for connected interval graphs G without an apex. Furthermore, we show how to define graphs that are isomorphic to the graph L_G from Section 7.2 in STC+C. In particular, this will enable us to prove, in Section 7.5, that an isomorphic copy of L_G on the number sort is STC+C-definable.

Let $G = (V, E)$ be a connected interval graph without an apex. Then G contains more than one max clique. Let $\mathfrak{P}_G$ be the set of all maximal proper subsets $\mathcal{C}$ of $\mathcal{M}_G$ with the property that for any $B \in \mathcal{M}_G \setminus \mathcal{C}$ we have $B \cap C = B \cap C'$ for all $C, C' \in \mathcal{C}$. We must have $|\mathfrak{P}_G| \geq 3$ since G is connected and no vertex may be included in all max cliques of G. Furthermore, if $\mathcal{C}, \mathcal{C}' \in \mathfrak{P}_G$ and $\mathcal{C} \neq \mathcal{C}'$, then $\mathcal{C} \cap \mathcal{C}' = \emptyset$. To see this, suppose that $D \in \mathcal{C} \cap \mathcal{C}'$. Then $B \cap A = B \cap D = B \cap C$ for all $A, C \in \mathcal{C} \cup \mathcal{C}'$ and $B \notin \mathcal{C} \cup \mathcal{C}'$. So as $|\mathfrak{P}_G| \geq 3$, $\mathcal{C} \cup \mathcal{C}'$ is a proper subset of $\mathcal{M}_G$ satisfying the above property, which contradicts the maximality of $\mathcal{C}$ and $\mathcal{C}'$. We conclude that $\mathfrak{P}_G$ is a partition of $\mathcal{M}_G$.

For each $\mathcal{C} \in \mathfrak{P}_G$ with $|\mathcal{C}| \geq 2$ we define $S_{\mathcal{C}} = \bigcup \mathcal{C} \setminus \bigcup (\mathcal{M}_G \setminus \mathcal{C})$. The correspondence in names to the modules $S_{\mathcal{C}}$ as defined in Lemma 7.6 is intended, of course, and makes sense since the sets $\mathcal{C} \in \mathfrak{P}_G$ enjoy the same interaction properties with the rest of the graph as maximal sets of $\prec_M$-incomparable max cliques (cf. Lemma 7.6).

We can now define the modules $\mathcal{W}_G$ mentioned in Section 7.2 for connected interval graphs G without an apex. We let $\mathcal{S} := \{S_{\mathcal{C}} \mid \mathcal{C} \in \mathfrak{P}_G \text{ with } |\mathcal{C}| \geq 2\}$, and define

$$\mathcal{W}_G := \mathcal{S} \cup \bigcup_{v \in V \setminus \bigcup \mathcal{S}} \{\{v\}\}.$$

From the fact that $\mathfrak{P}_G$ is a partition of $\mathcal{M}_G$, we conclude that $\mathcal{W}_G$ forms a partition of V, whereby inducing the equivalence relation $\sim_G$ on V. In the following, we call this equivalence relation alternatively $\sim_{\mathfrak{P}_G}$.

We are going to construct STC+C-definable graphs isomorphic to L_G. Let Z_M be the max clique which is $\prec_M$-maximal in G_M. Now we forget about $\prec_M$ and consider $\prec_{Z_M}$ on G_M. We write

$$L_M := G_M / {\sim_{Z_M}^{G_M}} = (V(G_M) / {\sim_{Z_M}^{G_M}}, E(G_M) / {\sim_{Z_M}^{G_M}})$$

with $E(G_M) / {\sim_{Z_M}^{G_M}} = \{(u / {\sim_{Z_M}^{G_M}}, v / {\sim_{Z_M}^{G_M}}) \mid (u, v) \in E(G_M)\}$. Lemma 7.7 implies again that $\prec_{Z_M}$ induces a linear order on the max cliques of L_M.

Lemma 7.8. *Let G be a connected interval graph that does not contain an apex, and let $M_1, \ldots, M_k$ be its possible ends. Then all of the graphs L_{M_l}, $l \in [k]$, are isomorphic to L_G and we may partition $[k]$ into at most two sets Q, Q' so that $(L_{M_i}, \prec_{Z_{M_i}})$ and $(L_{M_j}, \prec_{Z_{M_j}})$ are order isomorphic whenever $i, j \in Q$ or $i, j \in Q'$.*

Proof. Equivalence relation $\sim_{\mathfrak{P}_G}$ does the same as $\sim_M^G$, only that it is based on $\mathfrak{P}_G$ instead of the (finer) partition of max cliques induced by a strict weak ordering $\prec_M$.

Our goal is to show that each L_M with $M \in \{M_1, \ldots, M_k\}$ is isomorphic to $G / {\sim_{\mathfrak{P}_G}}$. For this it is enough to show that the concatenation of equivalence relation $\sim_M^G$ with $\sim_{Z_M}^{G_M}$ is equal to $\sim_{\mathfrak{P}_G}$. Whenever $\mathcal{C} \in \mathfrak{P}_G$ and $M \notin \mathcal{C}$, Lemma 7.5 implies that the max cliques in

$\mathcal{C}$ are $\prec_M$-incomparable. As the sets in $\mathfrak{P}_G$ were chosen to be maximal, $\mathcal{C}$ is also a maximal set of $\prec_M$-incomparables (Lemma 7.6). It follows that $\sim_{\mathfrak{P}_G}$ is equal to $\sim_M^G$ on $\bigcup_{M \notin \mathcal{C} \in \mathfrak{P}_G} \mathcal{C}$.

When forming $G_M = G/_{\sim_M^G}$, each maximal set of $\prec_M$-incomparable max cliques $\mathcal{C}$ is replaced by the max clique $M_{\mathcal{C}} = \{v/_{\sim_M^G} \mid v \in \bigcup \mathcal{C}\}$. Note that this is also true when $\mathcal{C}$ consists of just one max clique. As a result, $\mathfrak{P}_G$ induces a partition $\mathfrak{P}_M$ of the max cliques of G_M. Also, if $\mathcal{C}_M$ is the cell of $\mathfrak{P}_M$ which contains M, then $\mathcal{C}_M$ is the only cell of $\mathfrak{P}_M$ which is possibly not a singleton. As $|\mathfrak{P}_M| \geq 3$, $Z_M \notin \mathcal{C}_M$.

The final step is to show that $\sim_{\mathfrak{P}_M}$ equals $\sim_{Z_M}^{G_M}$ on G_M. If $v/_{\sim_M^G}$ is a vertex of G_M and $v/_{\sim_M^G}$ is an equivalence class of $\sim_M^G$ with $|v/_{\sim_M^G}| > 1$, then $v/_{\sim_M^G}$ is only contained in one max clique of G_M. Hence, $\mathfrak{P}_M$ inherits from $\mathfrak{P}_G$ the property that it partitions the max cliques $\mathcal{M}_{G_M}$ of G_M into maximal sets $\mathcal{C}$ so that for any $B \in \mathcal{M}_{G_M} \setminus \mathcal{C}$ we have $B \cap C = B \cap C'$ for all $C, C' \in \mathcal{C}$. Arguing analogously as above, it follows that $\sim_{\mathfrak{P}_M}$ equals $\sim_{Z_M}^{G_M}$. Therefore, $\sim_{\mathfrak{P}_G}$ is equal to the concatenation of $\sim_M^G$ with $\sim_{Z_M}^{G_M}$ and L_M is isomorphic to L_G. This proves the first part of the lemma.

To see the second part, observe that $\prec_{Z_M}$ induces a linear order on L_M's max cliques. This is true for all $M \in \{M_1, \ldots, M_k\}$, so whenever N is a possible end of L_M, then $\prec_N$ linearly orders the max cliques of L_M. Thus, L_M has two possible ends which correspondingly induce two orders on the max cliques and vertices of $G/_{\sim_{\mathfrak{P}_G}}$. $\qquad\square$

7.5. Canonising L_G.

Before showing how to use the modular decomposition tree for canonising interval graphs $G = (V, E)$ in our logic, let us take a look at how to define a canonical copy of L_G in STC+C.

From the fact that G is an interval graph, it is not hard to see that L_G is an interval graph, too. Furthermore, notice that, if A is a max clique of G, then

$$A_{L_G} := \{v/_{\sim_G} \mid v \in A\}$$

is a max clique of L_G, and that all max cliques of L_G are of this form.

Lemma 7.9.
(1) *There are STC+C-formulae $\varphi_\sim$, φ_L such that for all interval graphs G, $\varphi_\sim$ defines the equivalence relation $\sim_G$, and φ_L the edge relation of the graph L_G.*
(2) *Let G be a connected graph without any apices. If L_G has $m > 1$ max cliques, then there exist exactly two linear orderings of L_G's max cliques, each the reverse of the other. There is an STC+C-formula that defines all pairs of tuples $(u, v), (u', v') \in V^2$ such that $(u, v), (u', v')$ represent max cliques A, M of G, M is a possible end of G, and A_{L_G} appears within the first $\lfloor \frac{m}{2} \rfloor$ max cliques of L_G with respect to $\prec_{Z_M}$.*
(3) *There is an STC+C-formula that, for all interval graphs G, defines an isomorphic copy of L_G on the number sort.*

Proof. Let us start by showing property *(1)*. If G is not connected or G contains an apex, then $\sim_G$ is STC-definable. If G is connected and does not contain an apex, then for each possible end M of G the concatenation of equivalence relation $\sim_M^G$ with $\sim_{Z_M}^{G_M}$ is equal to $\sim_G$ (Lemma 7.8). The STC+C-definability of equivalence relation $\sim_G$ is a direct consequence of the STC+C-definability of the possible ends M and the equivalence relation $\sim_M^G$, Lemma 7.7, which allows us to define max clique Z_M, and the STC+C-definability of $\sim_{Z_M}^{G_M}$.

We do not define the graph L_G explicitly, but rather implicitly within G. That is, we do not single out a representative of each equivalence class $v/_{\sim_G}$ of $\sim_G$, but treat all vertices in $v/_{\sim_G}$ as representatives of $v/_{\sim_G}$. Notice that, since all equivalence classes of $\sim_G$ are modules of G, the edge relation of L_G can be defined as the set of all edges of G between vertices in different equivalence classes.

To show Property *(2)*, recall that by Lemma 7.8 there are exactly two linear orderings of L_G's max cliques, each the reverse of the other. By Property *(1)*, we can define L_G, and for a possible end M of L_G we can define the linear order $\prec_{Z_M}$ (Lemma 7.3). Hence, given max clique A, we can define A_{L_G} and associate the linear order with A where A_{L_G} appears within the first $\lfloor \frac{m}{2} \rfloor$ max cliques of L_G.

Property *(3)* is easy to see for graphs that are not connected or contain an apex. For connected interval graphs G that do not have any apices, Property *(3)* follows directly from Section IV.B in [19], where the author shows that there is an STC+C-formula that defines an ordered copy of G on the number sort if there is a max clique M of G such that $\prec_M$ is a linear order on G's max cliques.   □

According to the preceding lemma we can define an isomorphic copy of L_G on the number sort. In the following, we denote this copy by $\mathcal{K}(L_G)$.

7.6. The Coloured Modular Decomposition Tree.
To obtain a complete invariant of an interval graph $G = (V, E)$, we construct a refinement of the modular decomposition tree, the coloured modular decomposition tree, in this section.

Let us consider the modular decomposition tree $T(G)$ of an interval graph G. We call a module $W \in V(T(G))$ a *decomposition module* if $W = V$, or $|W| > 1$ and $G[W^*]$ is a connected graph, where W^* is the parent of W in $T(G)$. All modules W where $G[W^*]$ is not connected are called *component modules*. We let $\mathcal{W}_G^{dec}$ be the set of all decomposition modules and $\mathcal{W}_G^{con}$ be the set of all component modules occurring in the modular decomposition tree of G.

Let $P' := \{(M, n) \mid M \in \mathcal{M}_G, n \in [|V|]\}$. Recall the definition of the *span* of a vertex from Section 7.3, and that it is STC+C-definable. For each $(M, n) \in P'$, define $V_{M,n}$ as the set of vertices of the connected component of $G[\{v \in V \mid \text{span}(v) \le n\}]$ which intersects with M (if non-empty), and let $G_{M,n} := G[V_{M,n}]$. Now let P be the set of those $(M, n) \in P'$ for which the following properties are satisfied:

(1) The number n is maximal among those n' with the property that $V_{M,n'} = V_{M,n}$.
(2) For all $m' > n$ where $V_{M,m'}$ is a module, $V_{M,n}$ is a subset of an equivalence class of $\sim_{G_{M,m'}}$ with more than one vertex, or there exists a vertex $a \in V_{M,m'} \setminus V_{M,n}$ that is an apex of $G_{M,m'}$.

Lemma 7.10. *If $(M, n) \in P$, then $V_{M,n}$ is a connected component of a decomposition module in $\mathcal{W}_G^{dec}$. Moreover, if D is a connected component of a decomposition module in the modular decomposition tree of G, then there is an $(M, n) \in P$ with $V_{M,n} = D$.*

Proof. Notice that for all modules W of G and all max cliques C of G with $C \cap W \ne \emptyset$ the set $W \cap C$ is a max clique of $G[W]$, and every max clique of $G[W]$ is of that form. Further, an easy induction shows that for all modules $W \in \mathcal{W}_G^{dec} \cup \mathcal{W}_G^{con}$ the following properties are satisfied:

(A) Let $C, C' \in \mathcal{M}_G$ be max cliques of G with $C \ne C'$ where $C \cap W \ne \emptyset$ and $C' \cap W \ne \emptyset$. Then for max cliques $C \cap W$, $C' \cap W$ of $G[W]$ we have $C \cap W \ne C' \cap W$.

(B) Let $\mathcal{C} := \{C \in \mathcal{M}_G \mid C \cap W \neq \emptyset\}$. Then for all $B \in \mathcal{M}_G \setminus \mathcal{C}$ and all $C, C' \in \mathcal{C}$ we have $B \cap C = B \cap C'$.

(C) For the set $\mathcal{C}$ from (B), $W = \bigcup_{C \in \mathcal{C}} V_{C,c}$ where $c := |\mathcal{C}|$ if W contains an apex and $c := |\mathcal{C}| - 1$ if W has no apices, and for each $C \in \mathcal{C}$ the set $V_{C,c}$ is a connected component of W.

In order to show Lemma 7.10, we also need the following properties:

Claim 1. If $V_{M,k}$ is a connected component of a decomposition module W of G, and $V_{M,k} \subsetneq V_{M,l}$ for an $l > k$, then $W \subsetneq V_{M,l}$.

Proof. Let k' be the maximum span of a vertex in W. Since $V_{M,k} \in \mathcal{W}_G^{dec} \cup \mathcal{W}_G^{con}$, we have $V_{M,k} = V_{M,k'}$ as a direct consequence of Property C. Thus, we can assume that $k \geq k'$. Further, we have $V_{M,l} \not\subseteq W$ as $V_{M,l} \subseteq W$ leads to a contradiction, because $V_{M,l}$ is connected and $V_{M,k} \subsetneq V_{M,l}$ is a connected component of W. Thus, $V_{M,l} \setminus W$ is non-empty. Since $V_{M,l}$ is connected and $V_{M,k} \subseteq V_{M,l} \cap W$, there must exist a vertex $v \in V_{M,l} \setminus W$ that is adjacent to a vertex in the non-empty set $V_{M,l} \cap W$. As W is a module, v is adjacent to all vertices in W. Therefore, $W \subseteq V_{M,l}$, because $\mathrm{span}(w) \leq k' \leq l$ for all vertices $w \in W$. $\square$

Claim 2. Let $(M, d) \in P'$ and $V_{M,d}$ be a module in $\mathcal{W}_G^{dec} \cup \mathcal{W}_G^{con}$. If $V_{M,d}$ is a clique, then there exists only one max clique $C \in \mathcal{M}_G$ with $C \cap V_{M,d} \neq \emptyset$.

Proof. Since $V_{M,d}$ is a clique, there must exist a max clique $B \in \mathcal{M}_G$ with $V_{M,d} \subseteq B$. Let us assume, there exists a max clique $B' \in \mathcal{M}_G$ different from B with $B' \cap V_{M,d} \neq \emptyset$. According to Property A we have $B \cap V_{M,d} \neq B' \cap V_{M,d}$ and therefore $B' \cap V_{M,d} \subsetneq V_{M,d}$. Since $V_{M,d}$ is a module, $B' \cup V_{M,d}$ is a clique, a contradiction to B' being a max clique. $\square$

Claim 3. Let $(M, d) \in P'$ and $V_{M,d}$ be a module in $\mathcal{W}_G^{dec} \cup \mathcal{W}_G^{con}$. Further, let $0 < d' < d$ be such that $V_{M,d'} \subsetneq V_{M,d}$, and let $A \neq \emptyset$ be the set of apices of $G_{M,d}$. Then $V_{M,d'} \subseteq V_{M,d} \setminus A$.

Proof. If $V_{M,d}$ is a clique, then according to Claim 2 max clique M is the only max clique in $\mathcal{M}_G$ with $M \cap V_{M,d} \neq \emptyset$. Thus, $V_{M,d} = V_{M,1}$ and there does not exist a d' with $0 < d' < d$ such that $V_{M,d'} \subsetneq V_{M,d}$.

Now let $V_{M,d}$ be not a clique. Further, let $\mathcal{C}$ be the set of max cliques $C \in \mathcal{M}_G$ with $C \cap V_{M,d} \neq \emptyset$ and $c := |\mathcal{C}|$. In the following we show that $a \in V_{M,d}$ is an apex of $G_{M,d}$ if and only if $\mathrm{span}(a) = c$. If $a \in V_{M,d}$ and $\mathrm{span}(a) = c$, then a is contained in every max clique of G that has a non-empty intersection with $V_{M,d}$. As every vertex in $V_{M,d}$ is contained in at least one max clique of G, which of course has a non-empty intersection with $V_{M,d}$, a is an apex of $G_{M,d}$. Now let a be an apex of $G_{M,d}$ and let us assume that there exists a max clique $C \in \mathcal{M}_G$ with $C \cap V_{M,d} \neq \emptyset$ and $a \notin C$. Apex a is adjacent to all vertices in $C \cap V_{M,d}$, and since $V_{M,d}$ is a module, a is also adjacent to all vertices in $C \setminus V_{M,d}$. Therefore, $C \cup \{a\}$ is a clique, which is a contradiction to C being a maximal clique of G.

From $\mathrm{span}(v) = c$ for all vertices $v \in A$, $\mathrm{span}(v) < c$ for all $v \in V_{M,d} \setminus A$ and $V_{M,d'} \subsetneq V_{M,d}$ it follows that $d' < c$. Consequently, $V_{M,d'} \subseteq V_{M,d} \setminus A$. $\square$

To proceed with the proof of Lemma 7.10, we first show that if D is a connected component of a decomposition module $W \in \mathcal{W}_G^{dec}$ and $M \in \mathcal{M}_G$ with $M \cap D \neq \emptyset$, then there is an $n \in \mathbb{N}$ such that $(M, n) \in P$ and $V_{M,n} = D$.

We proof this by induction on the depth of the modular decomposition tree: Clearly, if D is a connected component of decomposition module V (i.e., a connected component of G), then $D = V_{M,|V|}$ for a max clique M with $M \cap D \neq \emptyset$, and $(M, |V|) \in P$.

Now, let D be a component of module $W \in \mathcal{W}_G^{dec}$ with $W \neq V$. Let c be the number c' of max cliques of G intersecting with W if W contains an apex and $c' - 1$ if W has no apices. According to Property C, $V_{M,c} = D$. Let n be maximal with $V_{M,n} = V_{M,c}$. Then $(M, n) \in P'$ and $D = V_{M,n}$. Choosing (M, n) like that ensures that Property 1 is satisfied for (M, n).

It remains to show Property 2. Let $m' > n$ and let $V_{M,m'}$ be a module. According to Property 1 we have $V_{M,n} \subsetneq V_{M,m'}$. Thus, Claim 1 implies $W \subsetneq V_{M,m'}$.

First, let us assume there exists an apex a of $G_{M,m'}$. If there exists an apex of $G_{M,m'}$ in $V_{M,m'} \setminus W$, we have shown Property 2. Thus, let us assume all apices of $G_{M,m'}$, in particular a, are in W. Since W is a module and $a \in W$, the vertex sets $V_{M,m'} \setminus W$ and W must be completely connected. If W contains two vertices w, w' that are not adjacent, then in the minimal interval representation the interval of each vertex in $V_{M,m'} \setminus W$ has to intersect the intervals of both w and w'. Thus, the intervals of all vertices in $V_{M,m'} \setminus W$ intersect with each other and each vertex in $V_{M,m'} \setminus W$ is an apex, a contradiction. Let us assume W is a clique. Let W^* be the parent module of W in the modular decomposition tree of G. Since W is a decomposition module, $|W| > 1$ and W^* contains either an apex, or is connected and contains no apices. W^* cannot contain an apex, because then all vertices in W^* form a clique and W is not in $\mathcal{W}_{G[W^*]}$. If W^* is connected and contains no apices, then $W = S_{\mathcal{C}}$ for $\mathcal{C} \in \mathfrak{P}_{G[W^*]}$ where $\mathcal{C}$ is a set of max cliques of $G[W^*]$ with $|\mathcal{C}| \geq 2$ (see Section 7.4). As W is connected, $W = V_{M,n}$. According to Claim 2 there exists only one max clique C of G with $C \cap W \neq \emptyset$. Consequently, $C' := C \cap W^*$ is the only max clique in $G[W^*]$ with $C' \cap W \neq \emptyset$, a contradiction. Hence, W cannot be a clique.

Now let us assume that there does not exist an apex of $G_{M,m'}$. Thus, $\sim_{G_{M,m'}}$ is constructed as described in Section 7.4. Let W' be the parent module W^* of W in the modular decomposition tree of G if W^* is a decomposition module, or if W^* is a component module, let W' be the parent of module W^*. Then W' is a decomposition module. Further, let D' be the component of W' that contains W. Notice that no matter what set we chose for W', we have $D' = W^*$. According to Property C, there exists an $n' \in [|V|]$ such that $D' = V_{M,n'}$. Let n' be maximal with that property. Therefore, $W^* = V_{M,n'}$ and W^* is a component of a decomposition module. By inductive assumption we have $(M, n') \in P$. If $V_{M,n'} = V_{M,m'}$, then $V_{M,n}$ is a subset of equivalence class W of $\sim_{G_{M,m'}}$ with more than one vertex and we are done. Therefore, let us assume $V_{M,n'} \neq V_{M,m'}$. If $n' < m'$, then $V_{M,n} \subseteq W \subsetneq W^* = V_{M,n'} \subsetneq V_{M,m'}$. As (M, n') satisfies Property 2 and there does not exist an apex in $G_{M,m'}$, the set $V_{M,n'}$, and therefore also the set $V_{M,n} \subsetneq V_{M,n'}$, is a subset of an equivalence class of $\sim_{G_{M,m'}}$ with more than one vertex.

It remains to consider $m' < n'$ where $V_{M,n'} \neq V_{M,m'}$. Then $V_{M,n} \subseteq W \subsetneq V_{M,m'} \subsetneq V_{M,n'} = W^*$. If $W^* = V_{M,n'}$ contains an apex, then $W = V_{M,n'} \setminus A$ where A is the set of apices of $G_{M,n'}$. According to Claim 3, $V_{M,m'} \subseteq V_{M,n'} \setminus A$. But this implies $V_{M,m'} \subseteq W$, a contradiction.

Finally, let us assume $W^* = V_{M,n}$ is connected and does not contain an apex. Then $W = S_{\mathcal{C}}$ for a $\mathcal{C} \in \mathfrak{P}_{G[W^*]}$ with $|\mathcal{C}| \geq 2$ where $\mathfrak{P}_{G[W^*]}$ is the set of all maximal proper subsets $\mathcal{C}'$ of $\mathcal{M}_{G[W^*]}$, the set of max cliques of $G[W^*]$, with the property that for any $B \in \mathcal{M}_{G[W^*]} \setminus \mathcal{C}'$ we have $C \cap B = C' \cap B$ for all $C, C' \in \mathcal{C}'$. For all $C \in \mathcal{M}_{G[W^*]}$ with $C \cap V_{M,m'} \neq \emptyset$ let $f(C)$ be the set $C \cap V_{M,m'}$. As $V_{M,m'}$ is a module, the set $\{f(C) \mid C \in \mathcal{M}_{G[W^*]}, C \cap V_{M,m'} \neq \emptyset\}$ is the set $\mathcal{M}_{G_{M,m'}}$ of max cliques of $G_{M,m'}$. Let $f(\mathcal{C})$ be the set $\{f(C) \mid C \in \mathcal{C}\}$. Then $f(\mathcal{C})$ is exactly the set of max cliques of $G_{M,m'}$ that have a non-empty intersection with W. Let

$f(C), f(C') \in f(\mathcal{C})$ and $f(B) \in \mathcal{M}_{G_{M,m'}} \setminus f(\mathcal{C})$. Then $f(C) \cap f(B) = f(C') \cap f(B)$, because $C \cap B = C' \cap B$ and therefore $(C \cap V_{M,m'}) \cap (B \cap V_{M,m'}) = (C' \cap V_{M,m'}) \cap (B \cap V_{M,m'})$. Further, $|f(\mathcal{C})| > 1$, since $|\mathcal{C}| > 1$ and for $C, C' \in \mathcal{C} \subseteq \mathcal{M}_{G[W^*]}$ with $C \neq C'$ we have $C \cap W \neq C' \cap W$ according to Property A. Consequently, $(C \cap V_{M,m'}) \cap W \neq (C' \cap V_{M,m'}) \cap W$ and $f(C) \neq f(C')$ for max cliques $f(C), f(C') \in f(\mathcal{C})$. We obtain that there exists a subset $f(\mathcal{C}')$ of $\mathcal{M}_{G_{M,m'}}$ with $f(\mathcal{C}) \subseteq f(\mathcal{C}')$ such that $f(\mathcal{C}') \in \mathfrak{P}_{G_{M,m'}}$. As there exists no max clique $f(B) \in \mathcal{M}_{G_{M,m'}} \setminus f(\mathcal{C}')$ with $f(B) \cap W \neq \emptyset$, $W \subseteq S_{f(\mathcal{C}')}$ and we have shown that $V_{M,n}$ is a subset of equivalence class $S_{f(\mathcal{C}')}$ of $\sim_{G_{M,m'}}$ with more than one vertex.

For the other direction, let $(M, n) \in P$, we need to show that $V_{M,n}$ is a component of a decomposition module. We prove this by induction on n. Clearly, this holds for $n = |V(G)|$, so let $n < |V(G)|$. Let p be minimal such that $p > n$ and $(M, p) \in P$. Since $(M, |V|) \in P$ such a number exists. By inductive assumption we know that $V_{M,p}$ is a component of a decomposition module. Thus, $V_{M,p}$ is a module occurring in $V(T(G))$, the vertices of the modular decomposition tree of G.

Since $(M, n) \in P$, (M, n) satisfies Property 2. Thus, $V_{M,n}$ is a subset of an equivalence class of $\sim_{G_{M,p}}$ with more than one vertex or there exists an apex of $G_{M,p}$ in $V_{M,p} \setminus V_{M,n}$.

Let $V_{M,n}$ be a subset of an equivalence class W of $\sim_{G_{M,p}}$ with more than one vertex. As $V_{M,p}$ is connected, the equivalence class W is a decomposition module. Let D be the connected component of W that contains $V_{M,n}$. If $V_{M,n} = D$, then $V_{M,n}$ is a component of a decomposition module and we are done. If $V_{M,n}$ is a proper subset of D, we obtain a contradiction to the choice of p, since we have already shown that for component D of decomposition module W there must exist an $m \in [|V|]$ such that $(M, m) \in P$ and $V_{M,m} = D$, and $n < m < p$.

Now let there be a vertex $a \in V_{M,p} \setminus V_{M,n}$ that is an apex of $G_{M,p}$. Let A be the set of apices of $G_{M,p}$. According to Claim 3 we have $V_{M,n} \subseteq V_{M,p} \setminus A$. Further, $|V_{M,p} \setminus A| = 1$ implies that $v \in V_{M,p} \setminus A$ is also an apex. Consequently, $|V_{M,p} \setminus A| > 1$. Therefore, we have either shown that $V_{M,n}$ is a component of equivalence class $V_{M,p} \setminus A$ of $\sim_{G_{M,p}}$ with more than one vertex or obtain a contradiction to the choice of p. $\qquad \square$

Corollary 7.11. *There is an* STC+C*-formula* $\varphi(x, y, z)$ *such that for all interval graphs* $G = (V, E)$, *all* $v, w \in V$, *and all* $n \in [|V|]$, *we have* $G \models \varphi[v, w, n]$ *iff* $M = \mathrm{N}^c(v) \cap \mathrm{N}^c(w)$ *is a max clique of* G *and* $(M, n) \in P$.

We are now ready to define the coloured modular decomposition tree. An illustration of the tree can be found in Figure 7.

Formally, the coloured modular decomposition tree is defined as $\mathcal{T} = \mathcal{T}_G = (V_{\mathcal{T}}, E_{\mathcal{T}})$, where the set $V_{\mathcal{T}}$ of nodes and the set $E_{\mathcal{T}}$ of edges of $\mathcal{T}$ is defined as follows. $V_{\mathcal{T}}$ is the union of the following sets:

- the set $\mathcal{V}$ of *component vertices* $v_{V_{M,n}}$, one for each set $V_{M,n}$ with $(M, n) \in P$,
- the set $\mathcal{A}$ of *arrangement vertices* $a_{\{\prec_Q\}, V_{M,n}}$ where $\{\prec_Q\}$ is the singleton set of the distinguished minimal order on $L_{G_{M,n}}$'s max cliques if $\mathcal{K}(L_{G_{M,n}})$ is not order isomorphic under its two linear orderings (recall the definition of $\mathcal{K}(L_{G_{M,n}})$ from Section 7.5). If $\mathcal{K}(L_{G_{M,n}})$ is order isomorphic under its two linear orderings, then max clique Q identifies an order $\prec_Q$, namely, the order where $Q_{L_{G_{M,n}}}$ occurs first (see Section 7.5 for the definition of $Q_{L_{G_{M,n}}}$). Q defines both orders if $Q_{L_{G_{M,n}}}$ is located in the middle. Thus, for each Q the set $\{\prec_Q\}$ is the set of orders containing either only one of the isomorphic orders or
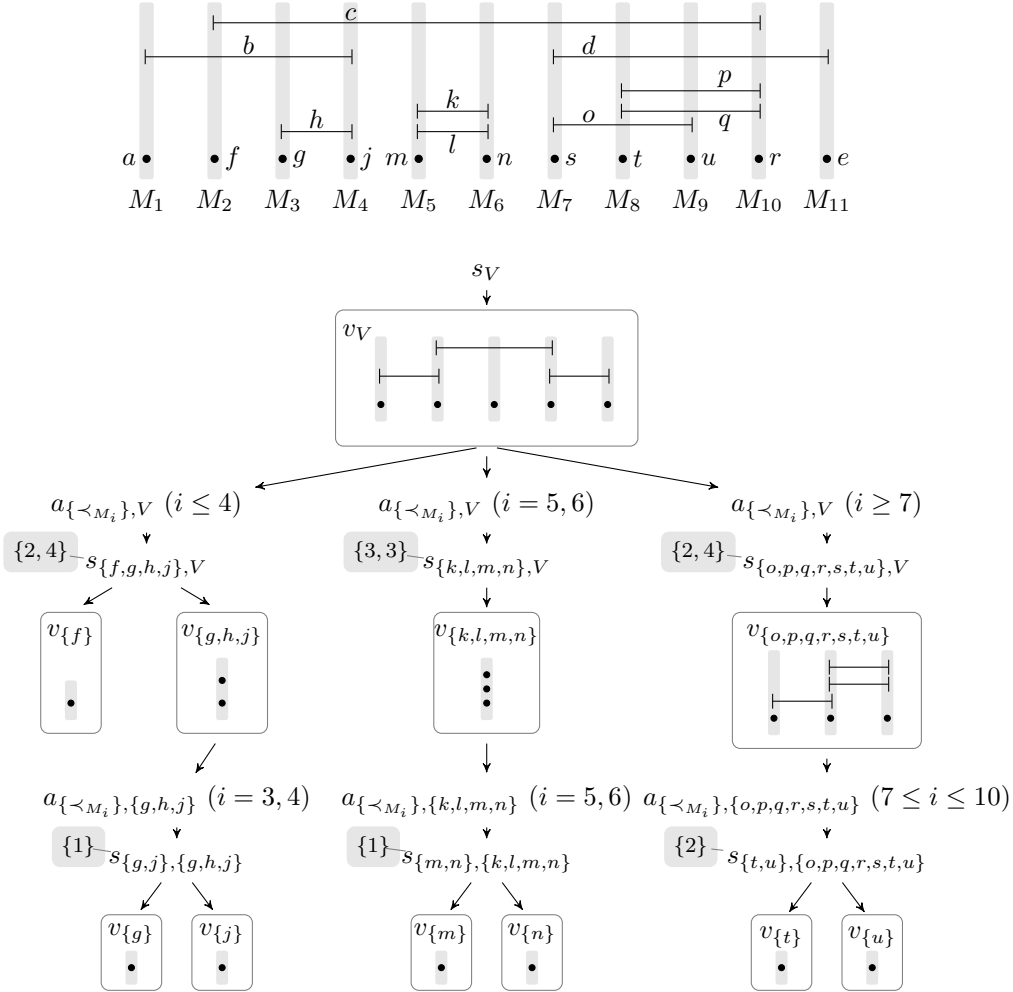
Figure 7: An interval graph and its coloured modular decomposition tree. Component
vertices v_U are represented together with the interval graph L_U labeling them.
The colours of module vertices are indicated in the gray fields next to them.

both. Consequently, for each set $V_{M,n}$ there are at most three arrangement vertices of
the form $a_{\{\prec_Q\},V_{M,n}}$.

• the set $\mathcal{S}$ of *module vertices* $s_{W_A,V_{M,n}}$ for which A is a max clique of G, and W_A is the
  vertex in $L_{G_{M,n}}$ (W_A is a module of $V_{M,n}$ with more than one vertex) that contains vertices
  of A, and

• $\{s_V\}$, where s_V is a special vertex acting as the root of $\mathcal{T}$.

We colour the vertices in $\mathcal{V}$ by assigning to each $v_{V_{M,n}} \in \mathcal{V}$ the ordered graph $\mathcal{K}(L_{G_{M,n}})$. The
vertices in $\mathcal{A}$ remain uncoloured and may therefore be exchanged by an automorphism of
$\mathcal{T}$ whenever their subtrees are isomorphic. Each $s_{W_A,V_{M,n}} \in \mathcal{S}$ is coloured with the multiset
of integers corresponding to the positions that the max clique $A_{L_{G_{M,n}}}$ takes in the orders
of $L_{G_{M,n}}$. The edge relation $E_{\mathcal{T}}$ of $\mathcal{T}$ is now defined in a straight-forward manner, with all
edges directed away from the root s_V.

- s_V is connected to all $v_{V_{M,n}} \in \mathcal{V}$ with $n = |V|$.
- Each $v_{V_{M,n}} \in \mathcal{V}$ is connected to all vertices in $\mathcal{A}$ of the form $a_{\{\prec_Q\},V_{M,n}}$ with $Q \cap V_{M,n} \neq \emptyset$. Therefore, $v_{V_{M,n}}$ is connected to at most three vertices.
- Each $a_{\{\prec_Q\},V_{M,n}} \in \mathcal{A}$ is connected to all those $s_{W_A,V_{M,n}} \in \mathcal{S}$ so that $\{\prec_Q\}$ is the set of orders of $L_{V_{M,n}}$ under which module $W_A \in V(L_{G_{M,n}})$ attains its minimal position, that is, for every max clique Q that intersects with a module W of $V_{M,n}$ with $|W| > 1$, vertex $a_{\{\prec_Q\},V_{M,n}} \in \mathcal{A}$ is connected to $s_{W_Q,V_{M,n}} \in \mathcal{S}$.
- Every $s_{W_A,V_{M,n}} \in \mathcal{S}$ is connected to those $v_{V_{M',n'}} \in \mathcal{V}$ for which $V_{M',n'}$ is a connected component of the module W_A, that is, for each max clique A the vertex $s_{W_A,V_{M,n}} \in \mathcal{S}$ is connected to $v_{V_{A,n'}} \in \mathcal{V}$ with $n' = \max\{m < n \mid (V_{A,m}) \in P\}$.

The point of the arrangement vertices $\mathcal{A}$ is to ensure that the order of submodules is properly accounted for. If our modular tree did not have such a safeguard, exchanging modules in symmetric positions might give rise to a non-isomorphic graph, but it would not change the tree, so $\mathcal{T}$ would be useless for the task of distinguishing between these two graphs.

We will later need STC+C-definability of this coloured tree. Thus, notice that the tree's vertices are equivalence classes, which are STC+C definable. Also the edge relation and the colours are STC+C-definable (Lemma 7.9).

Lemma 7.12 below shows that our modular trees are a complete invariant of interval graphs, so modular trees can be used to tell whether two interval graphs are isomorphic.

Lemma 7.12 ([18],[20]). *Let G and H be interval graphs. If their modular trees are isomorphic, then so are G and H.* $\qquad\square$

The graphs $L_{G_{M,n}}$ resemble the concept of *overlap components* used in [18] for the definition of a similar kind of modular tree. Overlap components are connected components of the subgraph of G in which only those edges are present for which the neighbourhood of neither endpoint is contained in the neighbourhood of the other (intuitively, their intervals *overlap*). It can be checked that overlap components and graphs $L_{G_{M,n}}$ only differ in the way they treat vertices that are contained in just one max clique: overlap components treat them as further modules (which they trivially are), the $L_{G_{M,n}}$ graphs directly put them into their unambiguous places. In [18] the authors show Lemma 7.12 for this similar kind of modular tree. A detailed proof of Lemma 7.12 can be found in [20].

7.7. **Total Preorder on Coloured Directed Trees.** We can make use of the STC+C-definable modular decomposition tree, and define a *total preorder* on the vertices of $\mathcal{T}_G$, that is, a linear order on the isomorphism classes of the (coloured) subtrees of $\mathcal{T}_G$ identified by its root vertices.

For our purposes, we define a *coloured directed tree* as a tuple $T = (V, E, L)$, where (V, E) is a directed tree and $L \subseteq V \times N(V)^2$ is a relation that assigns to each vertex $a \in V$ a colour $L_a := \{(m, n) \mid (a, m, n) \in L\}$. It is easy to bring the coloured modular decomposition tree into this form. For example, if a is a component vertex, say $v_{V_{M,n}}$, then L_a consists of all tuples (m, n), where (m, n) is an edge in the colour of a (i.e., an edge in the canon of $L_{V_{M,n}}$ by which a is coloured in $\mathcal{T}_G$). Furthermore, if a is a module vertex, say $s_{W_A,V_{M,n}}$, then L_a consists of all tuples (m, n), where m occurs n times in the colour of a. In all other cases, we simply leave L_a empty.

We let $\varphi_{\trianglelefteq}(x,y)$ be the formula such that for all coloured directed trees T, assignments α and $a, b \in V(T)$:

$$(T, \alpha) \models \varphi_{\trianglelefteq}[a, b] \iff L_a \text{ is lexicographically less than or equal to } L_b.$$

Then $\varphi_{\trianglelefteq}$ defines a total preorder $\trianglelefteq$ on the vertices of any coloured directed tree.

Let $\varphi_{\prec}(x,y)$ and $\varphi_{\cong}(x,y)$ be as defined in Section 4.2 and Section 4.1, respectively. If we identify each subtree of a directed tree with its root vertex, then the LREC$_=$-formula $\varphi_{\preceq}(x,y) := \varphi_{\prec}(x,y) \vee \varphi_{\cong}(x,y)$ defines a linear order $\preceq$ on the isomorphism classes of the subtrees of a directed tree.

We define a refinement $\preceq'$ of $\preceq$ by letting $v \prec' w$ whenever $v \triangleleft w$, or: $v \trianglelefteq w$ and $w \trianglelefteq v$ and $v \prec w$. It should be obvious how to modify the formula $\psi_{\preceq}(x,y)$ to an LREC$_=$-formula $\psi_{\preceq'}$ defining $\preceq'$.

7.8. Canonisation.

This section deals with the canonisation of interval graphs, that is, how to construct an LREC$_=$-formula $\kappa'(p,q)$ such that for each interval graph G we have $G \cong (\lVert V(G) \rVert, \kappa'[G; p, q])$. As a result we obtain the following:

Theorem 7.13. LREC$_=$ *captures* LOGSPACE *on the class of all interval graphs.*

We use the modular decomposition tree and the total preorder on its vertices for canonisation. We apply l-recursion on the modular decomposition tree, and as we have done for canonising trees we build the canon from the leaves to the root of the tree. Recursively, we construct the canon by first building the disjoint union of the canons of the components of submodules, then use the arrangement vertices to insert all submodules at the correct side and build the canon of the corresponding component of a module.

In the following we explain the canonisation procedure in more detail. The following lemma shows that it suffices to give an LREC$_=$-formula $\kappa(p,q)$ such that for every interval graph G we have $G \cong (\lVert V(G) \rVert, \kappa[\mathcal{T}_G; p, q])$. It follows from Lemma 6.6 and the fact that the coloured modular decomposition tree of an interval graph is STC+C-definable.

Lemma 7.14. *If there exists an* LREC$_=$*-formula* $\kappa(p,q)$ *such that for all interval graphs* G *we have* $G \cong (\lVert V(G) \rVert, \kappa[\mathcal{T}_G; p, q])$ *and* $\kappa[\mathcal{T}_G; p, q] \subseteq \lVert V(G) \rVert^2$, *then there also exists an* LREC$_=$*-formula* $\kappa'(p',q')$ *such that for all interval graphs* G, $G \cong (\lVert V(G) \rVert, \kappa'[G; p', q'])$.

Proof. As pointed out at the end of Section 7.6, the coloured modular decomposition tree of an interval graph G is definable in STC+C, and thus in LREC$_=$. That is, there are LREC$_=$-formulae $\theta_V(\bar{u})$, $\theta_{\approx}(\bar{u}, \bar{v})$, $\theta_E(\bar{u}, \bar{v})$ and $\theta_L(\bar{u}, \bar{q})$, where $\bar{u}, \bar{v}$ are compatible tuples and $\bar{q}$ is a tuple of number variables, such that for all interval graphs G and all assignments α,

- $\theta_{\approx}[G, \alpha; \bar{u}, \bar{v}]$ is an equivalence relation $\approx$,
- $\theta_V[G, \alpha; \bar{u}]/_{\approx}$ is the set of vertices of $\mathcal{T}_G$,
- $\theta_E[G, \alpha; \bar{u}, \bar{v}]/_{\approx} := \{(\bar{a}/_{\approx}, \bar{b}/_{\approx}) \mid (\bar{a}, \bar{b}) \in \theta_E[G, \alpha; \bar{u}, \bar{v}]\}$ is the edge relation of $\mathcal{T}_G$,
- and $\theta_L[G, \alpha; \bar{u}, \bar{q}]/_{\approx}$ is the colour-relation of the modular decomposition tree $\mathcal{T}_G$.

We now apply Lemma 6.6 with the transduction $\Theta = (\theta_V(\bar{u}), \theta_{\approx}(\bar{u}, \bar{v}), \theta_E(\bar{u}, \bar{v}), \theta_L(\bar{u}, \bar{q}))$ to obtain an LREC$_=$-formula $\kappa^{-\Theta}(\bar{p}', \bar{q}')$ such that for all $\bar{m}, \bar{n} \in N(G)^{|\bar{u}|}$, $G \models \kappa^{-\Theta}[\bar{m}, \bar{n}]$ iff $\langle \bar{m} \rangle_G, \langle \bar{n} \rangle_G \in N(\Theta[G])$ and $\Theta[G] \models \kappa[\langle \bar{m} \rangle_G, \langle \bar{n} \rangle_G]$. Note that $\Theta[G] = \mathcal{T}_G$. As $\kappa[\mathcal{T}_G; p, q] \subseteq \lVert V(G) \rVert^2$, the condition $\langle \bar{m} \rangle_G, \langle \bar{n} \rangle_G \in N(\Theta[G])$ can be replaced by $\langle \bar{m} \rangle_G, \langle \bar{n} \rangle_G \in N(G)$. Hence, the tuples $\bar{p}', \bar{q}'$ of number variables in $\kappa^{-\Theta}$ can be identified with single number variables p', q', which yields the desired formula $\kappa'(p', q')$. $\square$

In general, the canonisation procedure is similar to the one of directed trees. To apply l-recursion we use a graph $\mathtt{G} = (\mathtt{V}, \mathtt{E})$ with labels $\mathtt{C}(v) \subseteq \mathbb{N}$ for all $v \in \mathtt{V}$. We let $\mathtt{V} := V(\mathcal{T}_G) \times N(\mathcal{T}_G)^2$ be the vertices of $\mathtt{G}$ and for all component vertices $v_{V_{M,n}} \in \mathcal{V}$, $(v_{V_{M,n}}, p, q) \in V$ stands for "$(p, q) \in X_{v_{V_{M,n}}}$?", where $X_{v_{V_{M,n}}}$ is the edge relation of an isomorphic copy $([|V_{M,n}|], X_{v_{V_{M,n}}})$ of $G_{M,n}$.

In the following we explain the edge relation $\mathtt{E}$ and labels $\mathtt{C}$ of graph $\mathtt{G}$.

Edges introduced by module vertices.
In $\mathcal{T}_G$, each vertex $s_{W_A, V_{M,n}} \in \mathcal{S}$ is connected to those $v_{V_{M',n'}} \in \mathcal{V}$ for which $V_{M',n'}$ is a connected component of the module W_A. Thus, we can use the available total preorder $\prec'$ on the children of $s_{W_A, V_{M,n}}$ (cf. Section 7.7) to construct the canon of the disjoint union of the children's canons from the canons of the children. For a vertex $s \in \mathcal{S}$ and a child $v := v_{V_{M,n}} \in \mathcal{V}$ of s, let D_v be the set of all children v' of s with $v' \prec' v$, and e_v be the number of children v' of s defining modules isomorphic to $V_{M,n}$ (i.e., $v' \preceq' v$ and $v \preceq' v'$). For all $p, q \in N(\mathcal{T}_G)^2$ and all $i \in [0, e_v - 1]$, we let $\bar{a} := (s, p_{v,i} + p, p_{v,i} + q)$ have an edge to (v, p, q) where $p_{v,i} := \sum_{v_{V_{M',n'}} \in D_v} |V_{M',n'}| + i \cdot |V_{M,n}|$ and define $\mathtt{C}(\bar{a}) = \{e_v\}$. Notice that here we can have an in-degree greater than 1.

Edges introduced by arrangement vertices.
Let us consider a vertex $a_{\{\prec_Q\}, V_{M,n}} \in \mathcal{A}$. Its children in $\mathcal{T}_G$ are vertices $s_{W_A, V_{M,n}}$ for specific submodules of the module $V_{M,n}$, and we need to integrate the canons of them into the canon $\mathcal{K}(L_{V_{M,n}})$ of $L_{V_{M,n}}$. The canon $\mathcal{K}(L_{V_{M,n}})$ is $\mathsf{STC+C}$-definable (Lemma 7.9) and we assume it to be assigned to the first part $[1, |V(L_{V_{M,n}})|]$ of the number sort. Notice that on the number sort we have a distinguished ordering $<_N$ of the max cliques.

If $a_{\{\prec_Q\}, V_{M,n}} \in \mathcal{A}$ has no sibling, then we have a distinguished order of the max cliques of $L_{V_{M,n}}$, and we can integrate each canon of a submodule into $\mathcal{K}(L_{V_{M,n}})$ according to the colour of its vertex s. By integrating a submodule, we mean the following: We first sum up the size of $\mathcal{K}(L_{V_{M,n}})$ and the sizes of all submodules defined by children of $a_{\{\prec_Q\}, V_{M,n}}$ with smaller colours, and increase each vertex of the canon of the submodule by this number. Further, in the canon $\mathcal{K}(L_{V_{M,n}})$ we want to replace the smallest vertex z that lies in the max clique that is at the position defined by the colour of s and in no other max clique by the modified canon of the submodule. In order to do that, we add an edge between all vertices that are adjacent to z and all vertices of the modified canon of the submodule. For $a_{\{\prec_Q\}, V_{M,n}}$, we define the out-going edges of $a_{\{\prec_Q\}, V_{M,n}}$ in $\mathtt{G}$ such that, if X denotes the relation defined by the final $\mathsf{LREC_=}$-formula, the graph with edge relation $\{(p, q) \in N(T_G)^2 \mid ((a_{\{\prec_Q\}, V_{M,n}}, p, q), \ell) \in X$ for large enough $\ell\}$ consists of the modified canons of the submodules and all new edges. Note that we have not yet removed the replaced vertices.

If $a_{\{\prec_Q\}, V_{M,n}}$ has siblings, a single child, and the colour of the single child contains two equal positions, we know we have to insert the canon of its child in the middle (regarding the ordering of the max cliques) of $\mathcal{K}(L_{V_{M,n}})$. For such a vertex $a_{\{\prec_Q\}, V_{M,n}}$ we construct the edges of $\mathtt{G}$ so that we obtain the following graph on the number sort: We add the size of $\mathcal{K}(L_{V_{M,n}})$ to each vertex of the canon of the submodule, and add all edges that would be generated if we inserted the modified canon into the canon $\mathcal{K}(L_{V_{M,n}})$ replacing the smallest vertex in the middle max clique.

Now, let $a_{\{\prec_{Q_1}\}, V_{M,n}}$ and $a_{\{\prec_{Q_2}\}, V_{M,n}}$ be siblings where the colour of at least one child contains different positions. We determine their order with respect to the total preordering.

Say, $a_{\{\prec_{Q_1}\},V_{M,n}} \prec' a_{\{\prec_{Q_2}\},V_{M,n}}$. Then we want to integrate the submodules of $a_{\{\prec_{Q_1}\},V_{M,n}}$ all into the first half (regarding $<_N$) of the max cliques of canon $\mathcal{K}(L_{V_{M,n}})$, and the submodules of $a_{\{\prec_{Q_2}\},V_{M,n}}$ into the second half. Therefore, we create the edges of G in such a way that the graph on the number sort at vertex $a_{\{\prec_{Q_1}\},V_{M,n}}$ is as follows: Each child of vertex $a_{\{\prec_{Q_1}\},V_{M,n}}$ represents a certain submodule of $V_{M,n}$. We sum up the size of $\mathcal{K}(L_{V_{M,n}})$, the size of the submodule in the middle if it exists, and the sizes of all submodules defined by children of $a_{\{\prec_{Q_1}\},V_{M,n}}$ with smaller colours, and add this value to each vertex of the canon of this certain submodule. Finally, we insert each of these modified canons into the max clique specified by the smaller value contained in the colour of the corresponding vertex, in the same way we did before, and add all newly created edges to the modified canons of the submodules. For vertex $a_{\{\prec_{Q_2}\},V_{M,n}}$ we construct the graph on the number sort equivalently, only that we additionally add the sum of the sizes of all submodules defined by children of $a_{\{\prec_{Q_1}\},V_{M,n}}$ to the vertices of the canons of the submodules.

If $a_{\{\prec_{Q_1}\},V_{M,n}}$ and $a_{\{\prec_{Q_2}\},V_{M,n}}$ are equivalent with respect to the total preorder, we insert the submodules of $a_{\{\prec_{Q_i}\},V_{M,n}}$ for $i = 1,2$ at both sides. We position the submodules according to both values that are contained in their colours. Thus, if there is no submodule that belongs in the middle at vertex $a_{\{\prec_{Q_i}\},V_{M,n}}$, for $i = 1,2$, the edge relation of G almost enables us to define the canon of module $V_{M,n}$, except that we still need to remove the vertices that were replaced.

For each $a \in \mathcal{A}$ that fits in the last case, we let $\mathsf{C}(a,p,q) = \{2\}$, otherwise $\mathsf{C}(a,p,q) = \{1\}$, for all $p,q \in N(T_G)$. Note, that only in the last case, we obtain in-degrees larger than 1, that is, there the in-degree is 2.

Edges introduced by component vertices.
Let $v = v_{V_{M,n}} \in \mathcal{V}$. In the preceding step, we introduced edges for arrangement vertices $a_{\{\prec_Q\},V_{M,n}}$ so that, if X denotes the relation defined by the final $\mathsf{LREC}_=$-formula in an interval graph whose coloured modular decomposition tree is T_G, the graph with edge relation $\{(p,q) \in N(T_G)^2 \mid ((a_{\{\prec_Q\},V_{M,n}},p,q),\ell) \in X$ for large enough $\ell\}$ is almost a canon of $V_{M,n}$; we still need to insert $\mathcal{K}(L_{V_{M,n}})$, and remove the vertices of $\mathcal{K}(L_{V_{M,n}})$ that correspond to the submodules of $V_{M,n}$.

Recall from Lemma 7.9 that the canon $\mathcal{K}(L_{V_{M,n}})$ is $\mathsf{STC+C}$-definable. The set of vertices of $\mathcal{K}(L_{V_{M,n}})$ is $[1,|V(L_{V_{M,n}})|]$. Let R be the set of vertices that have to be removed from $\mathcal{K}(V_{M,n})$, so that the resulting graph plus the edges from $\{(p,q) \in N(T_G)^2 \mid ((a_{\{\prec_Q\},V_{M,n}},p,q),\ell) \in X$ for large enough $\ell\}$ is isomorphic to $V_{M,n}$. It is easy to define R by considering the different cases as we did above.

Let $f(r) := r - d_r$, where $d_r = |\{s \in R \mid s < r\}|$. Then, the contracted canon $\mathcal{Q} := \{(f(p),f(q)) \mid (p,q) \in \mathcal{K}(L_{V_{M,n}})\}$ is $\mathsf{STC+C}$-definable, and we assign it to the first part $[1,|V(L_{V_{M,n}})| - |R|]$ of the number sort. Thus, we set $\mathsf{C}(v,f(p),f(q)) = N(T_G)$ for all $(p,q) \in \mathcal{K}(L_{V_{M,n}})$. Furthermore, for each child $a \in \mathcal{A}$ of v, we include all edges from $(v,f(p),f(q))$ to (a,p,q) for all $p,q \in N(T_G) \setminus R$. Finally, for all $(p,q) \notin \mathcal{K}(L_{V_{M,n}})$ we set $\mathsf{C}(v,f(p),f(q)) = \{1\}$.

Finishing the construction.
In order to actually perform l-recursion we need sufficient "resources". Taking a look at the in-degrees, we notice that they are only larger than one when we treat isomorphic connected components while building the disjoint union, or when the graph $V_{M,n}$ is symmetric and we

insert the submodules twice at both sides. Either way, an incoming degree of d means that we insert d disjoint isomorphic copies into the graph on the number sort. Hence, it suffices to use a binary resource term.

Remark 7.15. It is possible to show that there is no $\mathsf{LREC+TC}[\{E\}]$-sentence φ such that for all connected interval graphs G_1, G_2 we have $G_1 \uplus G_2 \models \varphi$ if and only if $G_1 \cong G_2$. The proof is based on similar ideas as the proof of Theorem 5.1.

8. CONCLUSION

We introduce the new logics LREC and $\mathsf{LREC}_=$, extending first-order logic with counting by a recursion operator that can be evaluated in logarithmic space. By capturing $\mathsf{LOGSPACE}$ on trees and interval graphs, we obtain the first nontrivial descriptive characterisations of $\mathsf{LOGSPACE}$ on natural classes of unordered structures. It would be interesting to extend our results to further classes of structures such as the class of planar graphs or classes of graphs of bounded tree width.

The expressive power of $\mathsf{LREC}_=$ is not yet well-understood. For example, it is an open question whether directed graph reachability is expressible in $\mathsf{LREC}_=$, and even whether $\mathsf{LREC}_=$ has the same expressive power as $\mathsf{FP+C}$. (Of course assumptions from complexity theory indicate that the answer to both questions is negative.) It is also an open question whether reachability on undirected trees is expressible in plain LREC.

It is obvious that our capturing results can be transferred to nondeterministic logarithmic space NL by adding a transitive closure operator to the logic. However, it would be much nicer to have a natural "nondeterministic" variant of our limited recursion operator that allows it to express directed graph reachability and thus yields a logic that contains TC. We leave it as an open problem to find such an operator.

REFERENCES

[1] M. Beaudry and P. McKenzie. Circuits, matrices, and nonassociative computation. *J. Comput. Syst. Sci.*, 50(3):441–455, 1995.
[2] S. Buss, S. Cook, A. Gupta, and V. Ramachandran. An optimal parallel algorithm for formula evaluation. *SIAM J. Comput*, 21:755–780, 1992.
[3] J. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12:389–410, 1992.
[4] A. Chandra and D. Harel. Structure and complexity of relational queries. *J. Comput. Syst. Sci.*, 25:99–128, 1982.
[5] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer-Verlag, 1995.
[6] K. Etessami and N. Immerman. Tree canonization and transitive closure. *Inf. Comput.*, 157:2–24, 2000.
[7] R. Fagin. Generalized first–order spectra and polynomial–time recognizable sets. In R. M. Karp, editor, *Complexity of Computation, SIAM-AMS Proceedings 7*, pages 43–73, 1974.
[8] T. Gallai. Transitiv orientierbare Graphen. *Acta Math. Hungar.*, 18(1-2):25–66, 1967.
[9] P. C. Gilmore and A. J. Hoffman. A characterization of comparability graphs and of interval graphs. *Canad. J. Math.*, 16:539–548, 1964.
[10] E. Grädel, P.G. Kolaitis, L. Libkin, M. Marx, J. Spencer, M.Y. Vardi, Y. Venema, and S. Weinstein. *Finite Model Theory and Its Applications*. Springer-Verlag, 2007.
[11] M. Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. In *LICS*, 2010.
[12] M. Grohe, B. Grußien, A. Hernich, and B. Laubner. L-recursion and a new logic for logarithmic space. In *Proceedings of the 25th International Workshop on Computer Science Logic (CSL)*, pages 277–291, 2011.

[13] N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.

[14] N. Immerman. Expressibility as a complexity measure: results and directions. In *Structure in Complexity Theory Conference*, pages 194–202, 1987.

[15] N. Immerman. Languages that capture complexity classes. *SIAM J. Comput.*, 16:760–778, 1987.

[16] N. Immerman. *Descriptive Complexity*. Springer-Verlag, 1999.

[17] N. Immerman and E. Lander. Describing graphs: A first-order approach to graph canonization. In A. Selman, editor, *Complexity theory retrospective*, pages 59–81. Springer-Verlag, 1990.

[18] J. Köbler, S. Kuhnert, B. Laubner, and O. Verbitsky. Interval graphs: Canonical representation in logspace. In *ICALP (1)*, pages 384–395, 2010.

[19] B. Laubner. Capturing polynomial time on interval graphs. In *LICS*, pages 199–208, 2010.

[20] B. Laubner. *The Structure of Graphs and New Logics for the Characterization of Polynomial Time*. PhD thesis, Humboldt-Universität zu Berlin, 2011.

[21] L. Libkin. Logics with counting and local properties. *ACM Trans. Comput. Log.*, 1(1):33–59, 2000.

[22] L. Libkin. *Elements of Finite Model Theory*. Springer-Verlag, 2004.

[23] S. Lindell. A logspace algorithm for tree canonization. In *STOC*, pages 400–404, 1992.

[24] R. H. Möhring. *Graphs and Order*, volume 147 of *NATO ASI Series C, Mathematical and Physical Sciences*, pages 41–102. D. Reidel, 1984.

[25] O. Reingold. Undirected ST-connectivity in log-space. In *STOC*, pages 376–385, 2005.

[26] M.Y. Vardi. The complexity of relational query languages. In *STOC*, pages 137–146, 1982.