One-Dimensional Logic over Trees*

Emanuel Kieroński[1] and Antti Kuusisto[2]

1 **University of Wrocław, Poland**
kiero@cs.uni.wroc.pl
2 **University of Bremen, Germany**
kuusisto@uni-bremen.de

Abstract

A one-dimensional fragment of first-order logic is obtained by restricting quantification to blocks of existential quantifiers that leave at most one variable free. This fragment contains two-variable logic, and it is known that over words both formalisms have the same complexity and expressive power. Here we investigate the one-dimensional fragment over trees. We consider unranked unordered trees accessible by one or both of the descendant and child relations, as well as ordered trees equipped additionally with sibling relations. We show that over unordered trees the satisfiability problem is ExpSpace-complete when only the descendant relation is available and 2-ExpTime-complete with both the descendant and child or with only the child relation. Over ordered trees the problem remains 2-ExpTime-complete. Regarding expressivity, we show that over ordered trees and over unordered trees accessible by both the descendant and child the one-dimensional fragment is equivalent to the two-variable fragment with counting quantifiers.

1998 ACM Subject Classification F.4 Mathematical Logic and Formal Languages

Keywords and phrases satisfiability, expressivity, trees, fragments of first-order logic

Digital Object Identifier 10.4230/LIPIcs.MFCS.2017.64

1 Introduction

One-dimensional fragment of first-order logic, F_1, is obtained by restricting quantification to blocks of existential quantifiers that leave at most one variable free (as the logic is closed under negation and boolean operations one may also use blocks of universal quantifiers). It is not difficult to show that over general relational structures the satisfiability problem for F_1 is undecidable [8]. In such situations, there are two standard ways of regaining decidability. One can either try to impose some additional restrictions on the syntax of the considered logic or to restrict attention to some specific classes of structures. Both approaches have been tried in the context of F_1.

A nice syntactic restriction of F_1 which turns out to be decidable over general structures is called a *uniform* one-dimensional fragment, UF_1. It was introduced in [8] as a generalization of the two-variable fragment of first-order logic, FO^2, to contexts with relations of arity higher than two, e.g., databases. The readers interested in this variant are referred to [8], [11], [12] and a survey [14] which also reveals some connections with description logics.

Let us turn to the restricted classes of structures. There are two important first-choice options, well motivated in various areas of computer science, namely the class of words and the class of trees. F_1 over words and ω-words is investigated in [10]. The satisfiability

* E.K. was supported by the Polish National Science Centre grant No. 2016/21/B/ST6/01444. A.K. was supported by the ERC grant 647289 CODA.

problem is shown to be NExpTime-complete, exactly as in the case of FO^2 [7]. Moreover, over words F_1 and FO^2 turn out to share the same expressive power. The advantage of F_1 over FO^2 is that the former allows us to express some properties in a more natural way (and seems to be more succinct, which is however not formally proved). There are a few other related formalisms over words, worth mentioning here. In [7] it is shown that FO^2 is expressively equivalent to unary temporal logic, UTL, i.e., temporal logic with four navigational operators: *next state*, *somewhere in the future*, *previous state*, *somewhere in the past*. FO^2, however, is exponentially more succinct than UTL. The satisfiability problem for UTL is PSpace-complete. An extension of FO^2 by counting quantifiers, C^2, is shown to be NExpTime-complete over words in [5]. In fact, it is not difficult to observe that over words C^2 has the same expressive power as plain FO^2 (but, again, the former is more succinct). Another interesting extension of FO^2 over words, this time significantly increasing its expressive power, is an extension by the *between* predicate recently studied by in [13]. Its satisfiability problem is ExpSpace-complete.

Turning now to the class of trees, both FO^2 and C^2 retain a reasonable complexity, namely their satisfiability problems over trees are ExpSpace-complete. See [2] for the analysis of FO^2 over trees and [1] for its extension covering C^2. Regarding the expressive power, the situation depends on the type of the trees considered. In the case of *unordered* trees FO^2 cannot count and is less expressive than C^2. Over *ordered* trees both formalisms are equally expressive [1] and share the expressiveness with the navigational core of XPath temporal logic (cf. [15]). The importance of FO^2 and C^2 over trees is also justified by the fact that they are located close to the border between elementary and non-elementary, e.g., adding the third variable makes the satisfiability problem very hard: still decidable, but as shown in [18] necessarily with a non-elementary complexity.

In this paper we investigate the computational complexity and the expressive power of F_1 over trees. We consider finite unranked trees accessible by a few navigational signatures: just the descendant relation; just the child relation; both the descendant and the child relations; and, finally, the descendant relation, the child relation plus the next-sibling and following-sibling relations. Concerning the complexity of satisfiability, it depends on whether the child relation is present or not. With the child relation the satisfiability problem is 2-ExpTime-complete, and without it is ExpSpace-complete. To show the complexity results we perform some surgery on models leading to small model properties, and then design algorithms searching for such appropriate small models. Technically, we extend the approach from [4] used there in the context of FO^2. Roughly speaking we appropriately abstract the information about a node by its *profile* (an analogous notion is called *a full type* in [4]) and then we either contract nodes with the same profiles, or delete some nodes whose sufficiently many profiles are realized in a (fragment) of a model. Worth mentioning is that an orthogonal extension of the method from [4] is used in [1] in the context of C^2. In both cases the challenge is to carefully tune the notion of a profile (full type) in order to get the optimal complexity. Regarding expressivity, we argue that over ordered trees with all of the four navigational relations we consider, F_1 is expressively equivalent to C^2 and FO^2. We also show that over unordered trees equipped with both the descendant and the child relation F_1 is still equivalent to C^2 (but this time the latter is known to be more expressive than FO^2). While the former equivalence is rather easy to see (though slightly awkward to formally show), the latter is less obvious and more difficult to prove. In our expressivity studies we do not consider the cases of unordered trees accessible by only one of the descendant and the child relations. However, we conjecture that also under these scenarios F_1 is equivalent to C^2. We leave the related investigations for the full version of this paper.

In the case of trees, as in the case of F_1 over words, the advantage of F_1 over FO^2 and C^2 is that it allows to specify some properties in a more natural and elegant way. If we want to say that a tree contains some (especially not fully specified) pattern, consisting of more than two elements, we can just quantify an appropriate number of positions, and say how they should be labelled and related to each other. Expressing the same in FO^2 (if possible), and usually also in C^2, will very likely require some heavy recycling of the two available variables and a careful navigation over trees. Let us just recall a simple example from [10], which in fact does not even use the navigational relations. The formula $\exists xyz \bigwedge_{i=1}^{n}(P_i(x) \vee P_i(y) \wedge P_i(z))$ says that there are three points which together ensure that each unary properties $P_1, \ldots, P_n$ appears in a model. A reader is asked to check that complicated and long formulas arise when we try to express the same in FO^2 or even in C^2 over (ordered or unordered) trees.

The rest of the paper is organized as follows. In Section 2 we define the logic and structures we are interested in and introduce some tools which will be then used in the following sections. In Section 3 we perform some surgery on trees, which then, in Section 4, allows us to establish the exact complexity bounds under all the considered navigational scenarios. In Section 5 we consider expressivity issues, relating F_1 over trees with C^2, FO^2 and GF^2, and finally in Section 6 we conclude the paper.

2 Preliminaries

2.1 Trees and logics

We work with signatures of the form $\tau = \tau_0 \cup \tau_{bin}$, where τ_0 is a set of unary symbols and $\tau_{bin} \subseteq \{\downarrow, \downarrow_+, \rightarrow, \rightarrow^+\}$ is a set of *navigational* binary symbols. Over such signatures we consider the *one-dimensional* fragment of first-order logic F_1, that is the relational fragment in which quantification is restricted to blocks of existential quantifiers that leave at most one variable free. Formally, F_1 over relational signature τ and some countably infinite set of variables Var is the smallest set such that:

- $R\bar{x} \in \mathrm{F}_1$ for all $R \in \tau$ and all tuples $\bar{x}$ of variables from Var of the appropriate length,
- $x = y \in \mathrm{F}_1$ for all variables $x, y \in Var$,
- F_1 is closed under $\vee$ and $\neg$,
- if φ is an F_1 formula with free variables $x_0, \ldots, x_k$ then formulas $\exists x_0, \ldots, x_k \varphi$ and $\exists x_1, \ldots, x_k \varphi$ belong to F_1.

As usually, we can use standard abbreviations for other Boolean operations, like $\wedge, \rightarrow, \top$, etc., as well as for universal quantification. The length of a formula φ is measured in a natural way, and denoted $\|\varphi\|$. The *width* of a formula is the maximum of the numbers of free variables in its subformulas.

For a given formula φ we denote by $\tau_0(\varphi)$ the set of unary symbols that appear in φ. We write $\mathrm{F}_1[\tau_{bin}]$ to denote that the only binary symbols that are allowed are those from τ_{bin}.

We are interested in finite unranked tree structures, in which the interpretation of symbols from τ_{bin} is fixed: if available in the signature, $\downarrow$ is interpreted as the child relation, $\rightarrow$ as the right sibling relation, and $\downarrow_+$ and $\rightarrow^+$ as their respective transitive closures. If at least one of $\rightarrow$, $\rightarrow^+$ is interpreted in a tree then we say that this tree is *ordered*; in the opposite case we say that the tree is *unordered*. In this paper we investigate the four navigational signatures, namely, in the case of unordered trees, we consider accessing them only by the descendant relation ($\mathrm{F}_1[\downarrow_+]$), only by the child relation ($\mathrm{F}_1[\downarrow]$), and by both of them ($\mathrm{F}_1[\downarrow_+, \downarrow]$); in the case of ordered trees we consider just the full signature ($\mathrm{F}_1[\downarrow, \downarrow_+, \rightarrow, \rightarrow^+]$).

We use symbol $\mathfrak{T}$ (possibly with sub- or superscripts) to denote tree structures. For a given tree $\mathfrak{T}$ we denote by T its universe. If a is a node of $\mathfrak{T}$ then we denote by $\mathfrak{T}_a^{\downarrow}$ the subtree

of $\mathfrak{T}$ rooted at a, by $\mathfrak{T}_a^{\uparrow}$ the tree obtained from $\mathfrak{T}$ by removing all subtrees rooted at the children of a. Additionally, in the case of ordered trees we denote by $\mathfrak{T}_a^{\leftarrow}$ the substructure (which usually is not a tree) of $\mathfrak{T}$ generated by the nodes of subtrees rooted at a and all its left siblings (nodes a' such that $\mathfrak{T} \models a' \rightarrow^+ a$), and, symmetrically, we denote by $\mathfrak{T}_a^{\rightarrow}$ the substructure of $\mathfrak{T}$ generated by the nodes of subtrees rooted at a and all its right siblings.

2.2 Normal form

We adapt here the well known Scott normal form for FO^2 [16] to our purposes. We say that an $\mathrm{F}_1[\tau_{bin}]$ formula φ is in *normal form* if φ has the following shape:

$$\bigwedge_{1 \leq i \leq m_\exists} \forall y_0 \exists y_1 \dots y_{k_i} \varphi_i^\exists \wedge \bigwedge_{1 \leq i \leq m_\forall} \forall x_1 \dots x_{l_i} \varphi_i^\forall, \tag{1}$$

where $\varphi_i^\exists = \varphi_i^\exists(y_0, y_1, \dots, y_{k_i})$ and $\varphi_i^\forall = \varphi_i^\forall(x_1, \dots, x_{l_i})$ are quantifier-free. Please note that the width of φ is the maximum of the set $\{k_i + 1\}_{1 \leq i \leq m_\exists} \cup \{l_j\}_{1 \leq j \leq m_\forall}$. The following fact can be proved in a standard fashion, see, e.g., [6] for a more detailed exposition of the technique.

▶ **Lemma 1.** *For every $\mathrm{F}_1[\tau_{bin}]$ formula φ, one can compute in polynomial time an $\mathrm{F}_1[\tau_{bin}]$ formula φ' in normal form (over the signature extended by some fresh unary symbols) such that for trees of size (number of nodes) equal at least to the width of φ: (i) any model of φ can be expanded to a model of φ' by appropriately interpreting fresh unary symbols; (ii) any model of φ' restricted to the signature of φ is a model of φ.*

Proof. (Sketch) We successively replace innermost subformulas ψ of φ of the form $\exists y_1, \dots, y_k$ $\varphi(y_0, y_1, \dots, y_k)$ by atoms $P_\psi(y_0)$, where P_ψ is a fresh unary symbol, and axiomatize P_ψ using two normal form conjuncts: $\forall y_0 \exists y_1, \dots, y_k (P_\psi(y_0) \rightarrow \varphi(y_0, y_1, \dots, y_k))$ and $\forall y_0, y_1, \dots, y_k$ $(\neg\varphi(y_0, y_1, \dots, y_k) \vee P_\psi(y_0))$. ◀

Lemma 1 allows us, when dealing with satisfiability or when analysing the size and shape of models, to restrict attention to normal form formulas (models of size smaller than the width of the considered formula can be easily treated separately).

2.3 Types and profiles

In this subsection we prepare some notions useful in the rest of this paper.

2.3.1 Types

For $k \in \mathbb{N} \setminus \{0\}$ a *k-type* (or a *type of size k*) π over a signature $\tau = \tau_0 \cup \tau_{bin}$ is a set of literals over variables $x_1, \dots, x_k$ (often identified with a conjunction of its elements) such that

- for each $P \in \tau_0$ and $1 \leq i \leq k$ either Px_i or $\neg Px_i$ belongs to π
- for each $\rightleftharpoons \in \tau_{bin}$ and $1 \leq i, j, \leq k$, $i \neq j$ either $x_i \rightleftharpoons x_j$ or $\neg x_i \rightleftharpoons x_j$ belongs to π
- for each $1 \leq i < j \leq k$ the inequality $x_i \neq x_j$ belongs to π
- π is satisfiable in a tree, i.e., there exists a tree $\mathfrak{T}$ containing nodes $a_1, \dots, a_k$ such that $\mathfrak{T} \models \pi(a_1, \dots, a_k)$

In this paper we will only be interested in k-types over signatures containing $\downarrow_+$. If for some variable x_i and all $j \neq i$ we have that $x_i \downarrow_+ x_j \in \pi$ then we call x_i the *root* of π. If for some variable x_i and all $j \neq i$ either $x_j \downarrow_+ x_i \in \pi$ or $x_i \downarrow_+ x_j \notin \pi$ and $x_j \downarrow_+ x_i \notin \pi$ then we call x_i a *leaf* of π. Additionally for signatures containing horizontal relations: If for some variable x_i

and all $j \neq i$ either $x_i \rightarrow^+ x_j \in \pi$ or there is h such that $x_i \rightarrow^+ x_h, x_h \downarrow_+ x_j \in \pi$ then we call x_i the *leftmost element* of π. Analogously we define the *rightmost element* of π.

Note that a k-type may have at most one root, one leftmost element and one rightmost element, but many leaves. A *type* is a k-type for some $k \geq 1$.

We say that a tuple of distinct nodes $a_1 \ldots, a_k$ of a tree $\mathfrak{T}$ *realizes* a k-type π if $\mathfrak{T} \models \pi[a_1, \ldots, a_k]$. In this case we write $\text{type}^{\mathfrak{T}}(a_1, \ldots, a_k) = \pi$.

For a given k-type π and a sequence of variables $x_{i_1}, \ldots, x_{i_k}$ we denote by $\pi(x_{i_1}, \ldots, x_{x_k})$ the result of the simultaneous substitution $x_j \leftarrow x_{i_j}$ in π. Note that in this operation $i_1, \ldots, i_k$ is not required to be a permutation of $1, \ldots, k$.

2.3.2 Subtypes

Observe that a 1-type is completely determined by a subset of τ_0. We denote by $\pi \restriction x_i$ the 1-type obtained by restricting π to literals over x_i and then replacing in them x_i by x_1. More generally, for distinct indices $i_1, \ldots, i_l \in \{1, \ldots, k\}$ we denote by $\pi \restriction [x_{i_1} \ldots x_{i_l}]$ the l-type obtained by restricting π to literals over $x_{i_1}, \ldots, x_{i_l}$, and then replacing in them x_{i_j} by x_j (for $j = 1, \ldots, l$). We say that $\pi \restriction [x_{i_1} \ldots x_{i_l}]$ is a *subtype* of π; if $i_1 = 1$ then such a subtype is called an *initial subtype* of π. Initial subtypes may be formed with $l = k$ and are called *rearrangement* of π in this case. We say that a set $\mathcal{C}$ of types is *closed under initial subtypes* if for any k-type $\pi \in \mathcal{C}$ and any distinct $i_2, \ldots, i_l \in \{2, \ldots, k\}$, we have $\pi \restriction [x_1, x_{i_2}, \ldots, x_{i_l}] \in \mathcal{C}$.

2.3.3 Profiles

Profiles are intended to abstract the information about a node in a tree. Namely, they say what are the types of tuples (of some bounded size) containing the given element. For convenience we will separately store the types of tuples built of the nodes *above* and *below* the given element.

A *k-profile* over a signature τ (containing $\downarrow_+$) is a tuple $(\alpha, \mathcal{A}, \mathcal{B})$ such that:

- α is a 1-type,
- $\mathcal{A}$ is a set of types closed under initial subtypes, such that for all $\pi \in \mathcal{A}$: (i) π is of size at most k; (ii) $\pi \restriction x_1 = \alpha$ and (iii) x_1 is a leaf of π,
- $\mathcal{B}$ is a set of types closed under initial subtypes, such that for all $\pi \in \mathcal{B}$: (i) π is of size at most k; (ii) $\pi \restriction x_1 = \alpha$ and (iii) x_1 is the root of π.

Given a profile θ we will sometimes refer to its components as $\theta.\alpha$, $\theta.\mathcal{A}$ and $\theta.\mathcal{B}$.

In the case of the signature $\{\downarrow, \downarrow_+, \rightarrow, \rightarrow^+\}$ we also consider *horizontal* k-profiles, i.e., tuples of the form $(\alpha, \mathcal{A}, \mathcal{B}, \mathcal{A}_L, \mathcal{A}_R)$, extending k-profiles in such a way that:

- $\mathcal{A}_L$ is a set of types closed under the initial subtypes, such that for all $\pi \in \mathcal{A}_L$: (i) π is of size at most k, (ii) $\pi \restriction x_1 = \alpha$ and (iii) x_1 is the leftmost element of π,
- $\mathcal{A}_R$ is a set of types closed under the initial subtypes, such that for all $\pi \in \mathcal{A}_L$: (i) π is of size at most k, (ii) $\pi \restriction x_1 = \alpha$ and (iii) x_1 is the rightmost element of π,

By simple calculations we get:

▶ **Claim 2.** *For any navigational signature τ_{bin} we have:*

(i) *The number of k-types over $\tau = \tau_0 \cup \tau_{bin}$ is bounded exponentially in $|\tau_0|$ and k. In particular, there are $2^{|\tau_0|}$ 1-types.*

(ii) *The number of k-profiles and horizontal k-profiles over $\tau_0 \cup \tau_{bin}$ are bounded doubly exponentially in $|\tau_0|$ and k.*

We denote by $\mathrm{prof}_k^{\mathfrak{T}}(a)$ the k-profile *realized* by a in $\mathfrak{T}$, i.e., the profile $(\alpha, \mathcal{A}, \mathcal{B})$ such that:

- α is the 1-type of a,
- $\mathcal{A}$ is the set of types of size at most k realized by tuples $a_1, a_2, \ldots, a_l \in T_a^{\uparrow}$ such that $a_1 = a$,
- $\mathcal{B}$ is the set of types of size at most k realized by tuples $a_1, a_2, \ldots, a_l \in T_a^{\downarrow}$ such that $a_1 = a$.

An element $a \in T$ *realizes* a horizontal k-profile $(\alpha, \mathcal{A}, \mathcal{B}, \mathcal{A}_L, \mathcal{A}_R)$ if it realizes $(\alpha, \mathcal{A}, \mathcal{B})$ and

- $\mathcal{A}_L$ is the set of types of size at most k realized by tuples $a_1, a_2, \ldots, a_l \in T_a^{\leftarrow}$ such that $a_1 = a$,
- $\mathcal{A}_R$ is the set of types of size at most k realized by tuples $a_1, a_2, \ldots, a_l \in T_a^{\rightarrow}$ such that $a_1 = a$.

Note that in realized horizontal profiles it is also the case that both $\mathcal{A}_L$ and $\mathcal{A}_R$ are subsets of $\mathcal{A}$. Also all of $\mathcal{A}, \mathcal{B}, \mathcal{A}_L, \mathcal{A}_R$ are closed under initial subsets.

In the sequel, whenever a formula $\varphi \in \mathrm{F}_1[\tau_{bin}]$ is fixed we silently assume that all k-types, k-profiles, horizontal-k-profiles and all structures considered are over the signature $\tau_{bin} \cup \tau_0(\varphi)$.

Let us consider a k-profile $\theta = (\alpha, \mathcal{A}, \mathcal{B})$. We say that an l-type π $(1 \leq l \leq k)$ is *implicit* in θ if π is a member of $\mathcal{A}$ or $\mathcal{B}$ or if there are l_1-type $\pi_1 \in \mathcal{A}$, l_2-type $\pi_2 \in \mathcal{B}$, $l_1 + l_2 - 1 = k$, such that π is the unique l-type containing $\pi_1 \cup \pi_2(x_1, x_{l_1+1}, x_{l_1+2}, \ldots, x_{l_1+l_2-1})$, Note that if for some $2 \leq i \leq l_1$ we have $x_i{\downarrow_+}x_1 \in \pi_1$ then $x_i{\downarrow_+}x_j \in \pi$ for all $j \geq l_1$ and if $x_i{\downarrow_+}x_1 \notin \pi$ then also $x_j{\downarrow_+}x_1 \notin \pi$ for all $j \geq l_1$. Intuitively, an l-type π is implicit in θ if in any tree in which θ is realized by a node a there is a tuple of nodes starting with a realizing π.

Let φ be a normal form formula of width n. Given an n-profile θ one can easily see if its any realization in any tree $\mathfrak{T}$ have all the witnesses required by $\forall\exists...\exists$ conjuncts of φ, and if each tuple of nodes of $\mathfrak{T}$ containing a cannot violate any universal conjunct of φ. Formally, we say that an n-profile θ is *φ-admissible* if

1. for every conjunct of φ of the form $\forall y_0 \exists y_1 \ldots y_{k_i} \varphi_i^{\exists}(y_0, \ldots, y_{k_i})$ there is a k-type π $(k \leq k_i)$ implicit in θ and a function $h : \{y_1, \ldots, y_{k_i}\} \to \{x_1, \ldots, x_{l_i}\}$, such that $\pi \models \varphi_i^{\exists}[y_0/x_1, y_1/h(y_1), \ldots, y_{l_i}/h(y_{k_i})]$.
2. for every conjunct of φ of the form $\forall y_1 \ldots y_{l_i} \varphi_i^{\forall}(y_1, \ldots, y_{l_i})$ any k-type $(k \leq l_i)$ implicit in θ and any function $h : \{y_1, \ldots, y_{l_i}\} \to \{x_1, \ldots, x_k\}$ having x_1 in its image we have $\pi \models \varphi_i^{\forall}[y_1/h(y_1), \ldots, y_1/h(y_1), \ldots, y_{l_i}/h(y_{k_i})]$

It is then straightforward to see the following.

▶ **Lemma 3.** *Let φ be a normal form formula of width n. Then $\mathfrak{T} \models \varphi$ iff all n-profiles realized in $\mathfrak{T}$ are φ-admissible.*

In our decision procedures we will sometimes check admissibility of profiles. Since the number of types implicit in a profile is bounded polynomially this can be done (relatively) easily.

▶ **Claim 4.** *Given a normal form F_1 formula φ of width n and an n-profile θ it can be checked in nondeterministic polynomial time (in $\|\varphi\|$ and $|\theta|$) whether θ is φ-admissible.*

In the next section we perform some surgery on models of a normal form formula φ. Namely, we remove some nodes and sometimes change the connections among the remaining nodes. Observing that the n-profiles of the surviving nodes are not changed we will then be able to conclude (thanks to Lemma 3) that the resulting trees are still models of φ.

3 Pruning trees

We now show that when looking for models of a formula φ one can restrict attention to models with bounded depth and degree. We obtain an exponential bound on the depth in the case of signatures not containing $\downarrow$, and a doubly exponential bound when $\downarrow$ is present. The bound on the degree is exponential for unordered trees and doubly exponential for ordered trees. We remark that all the above bounds are essentially optimal.

3.1 Bounded paths

The crucial observation here is that one can remove the fragment of a model between two nodes having the same profile.

▶ **Lemma 5.** *Let τ_{bin} be any of the navigational signatures $\{\downarrow\}$, $\{\downarrow_+\}$, $\{\downarrow,\downarrow_+\}$, $\{\downarrow,\downarrow_+,\rightarrow,\rightarrow^+\}$. Let φ be a normal form $\mathrm{F}_1[\tau_{bin}]$ formula of width n. Let $\mathfrak{T} \models \varphi$ and let $a, b \in T$ be two nodes such that $\mathfrak{T} \models a\downarrow_+ b$ and $\mathrm{prof}_n^{\mathfrak{T}}(a) = \mathrm{prof}_n^{\mathfrak{T}}(b)$. Let $\mathfrak{T}'$ be the tree obtained from $\mathfrak{T}$ by replacing the subtree rooted at a by the subtree rooted at b. Then for any $c \in T'$ we have that $\mathrm{prof}_n^{\mathfrak{T}'}(c) = \mathrm{prof}_n^{\mathfrak{T}}(c)$. In consequence $\mathfrak{T}' \models \varphi$.*

Proof. Consider the case when c belongs to $\mathfrak{T}'^{\downarrow}_b$ (possibly $c = b$). Since $\mathfrak{T}'^{\downarrow}_b = \mathfrak{T}^{\downarrow}_b$ then in particular $\mathfrak{T}'^{\downarrow}_c = \mathfrak{T}^{\downarrow}_c$ and it is clear that $\mathrm{prof}_n^{\mathfrak{T}'}(c).\mathcal{B} = \mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{B}$.

To see that $\mathrm{prof}_n^{\mathfrak{T}'}(c).\mathcal{A} \subseteq \mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{A}$ consider any $\pi \in \mathrm{prof}_n^{\mathfrak{T}'}(c).\mathcal{A}$. Take a realization of π in $\mathfrak{T}'$ starting with c. Let $c, b_1, \ldots, b_k, a_1, \ldots, a_l$ be a list of all nodes of this realization, such that $b_1, \ldots, b_k \in T'^{\downarrow}_b$ and $a_1, \ldots, a_l \notin T'^{\downarrow}_b$. Let $\pi_0 = \mathrm{type}^{\mathfrak{T}'}(c, b_1, \ldots, b_k, a_1, \ldots, a_l)$. Note that π_0 is a rearrangement of π. Let $\pi'_0 = \mathrm{type}^{\mathfrak{T}'}(b, a_1, \ldots, a_l)$. Note that $\pi'_0 = \mathrm{type}^{\mathfrak{T}}(a, a_1, \ldots, a_l)$ and thus $\pi'_0 \in \mathrm{prof}_n^{\mathfrak{T}}(a).\mathcal{A} = \mathrm{prof}_n^{\mathfrak{T}}(b).\mathcal{A}$. It follows that π'_0 is realized in $\mathfrak{T}$ by $b, a'_1, \ldots, a'_l$ for some $a'_1, \ldots, a'_l \in T^{\uparrow}_b$. Observe that $\mathrm{type}^{\mathfrak{T}}(c, b_1, \ldots, b_k, a'_1, \ldots, a'_l) = \pi_0$ and thus $\pi_0 \in \mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{A}$. As $\mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{A}$ is closed under initial subtypes (and thus also rearrangements) it follows that $\pi \in \mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{A}$

For the opposite direction, consider now any $\pi \in \mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{A}$. Take a realization of π in $\mathfrak{T}$ starting with c. Let $c, b_1, \ldots, b_k, a_1, \ldots, a_l$ be a list of all nodes of this realization such that $b_1, \ldots, b_k \in T^{\downarrow}_b$ and $a_1, \ldots, a_l \notin T^{\downarrow}_b$. Let $\pi_0 = \mathrm{type}^{\mathfrak{T}}(c, b_1, \ldots, b_k, a_1, \ldots, a_l)$ Let $\pi'_0 = \mathrm{type}^{\mathfrak{T}}(b, a_1, \ldots, a_l)$. Note that $\pi'_0 \in \mathrm{prof}_n^{\mathfrak{T}}(b).\mathcal{A} = \mathrm{prof}_n^{\mathfrak{T}}(a).\mathcal{A}$ and thus $a, a'_1, \ldots, a'_l$ realize π'_0 in $\mathfrak{T}'$ for some $a'_1, \ldots, a'_l$. Now $\mathrm{type}^{\mathfrak{T}'}(b, a'_1, \ldots, a'_l) = \pi'_0$ and $\mathrm{type}^{\mathfrak{T}'}(c, b_1, \ldots, b_k, a'_1, \ldots, a'_l) = \pi_0$. Thus $\pi_0 \in \mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{A}$ and since π is a rearrangement of π_0 then also $\pi \in \mathrm{prof}_n^{\mathfrak{T}}(c).\mathcal{A}$.

The case when $c \in T'^{\uparrow}_b$ can be analysed analogously. Since we have shown that all profiles of nodes in $\mathfrak{T}'$ are realized in $\mathfrak{T}$ it follows by Lemma 3 that $\mathfrak{T}' \models \varphi$. ◀

Having proved Lemma 5 we can now obtain the desired bounds.

▶ **Corollary 6.**

(i) *Every satisfiable $\mathrm{F}_1[\downarrow]$, $\mathrm{F}_1[\downarrow,\downarrow_+]$ or $\mathrm{F}_1[\downarrow,\downarrow_+,\rightarrow,\rightarrow^+]$ normal form formula φ has a model whose vertical root-to-leaf paths are bounded doubly exponentially in $\|\varphi\|$ by a fixed function $\mathfrak{f}_d$.*

(ii) *Every satisfiable $\mathrm{F}_1[\downarrow_+]$ normal form formula φ has a model whose vertical root-to-leaf paths are bounded exponentially in $\|\varphi\|$ by a fixed function $\mathfrak{f}_s$.*

Proof. Take a model $\mathfrak{T} \models \varphi$. If there are nodes a, b meeting conditions of Lemma 5 then replace the subtree rooted at a by the subtree rooted at b. Repeat this operation until all root-to-leaf paths realize only distinct n-profiles. Let $\mathfrak{T}^*$ be the eventually obtained tree. By Lemma 5 we have $\mathfrak{T}^* \models \varphi$. To see (i) just recall that by Claim 2 (ii) the number of distinct

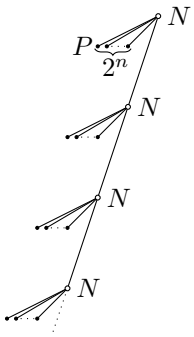

Figure 1 Enforcing doubly exponential path in $\mathrm{F}_1[\downarrow]$.

n-profiles over $\{\downarrow\}$, $\{\downarrow, \downarrow_+\}$ or $\{\downarrow, \downarrow_+, \rightarrow, \rightarrow^+\}$ is bounded doubly exponentially in $\tau_0(\varphi)$ and n and thus also in $\|\varphi\|$. For (ii) take any vertical root-to-leaf path p of $\mathfrak{T}^*$ and consider any 1-type α realized on this path. Let $a_1, \ldots, a_k$ be the list of all nodes of p realizing α, $\mathfrak{T}^* \models a_i \downarrow_+ a_j$ for $i < j$. Let $(\alpha, \mathcal{A}_i, \mathcal{B}_i)$ be the n-profile (over $\{\downarrow_+\}$) of a_i for $1 \leq i \leq k$. We observe that for $i < j$ we have $\mathcal{A}_i \subseteq \mathcal{A}_j$ and, $\mathcal{B}_i \supseteq \mathcal{B}_j$. Let us explain the former of these two inclusions. Take any k-type $\pi \in \mathcal{A}_i$ and let $a_i, b_2, \ldots, b_k$ be its realization. Note that the nodes $b_2, \ldots, b_k$ are related by $\downarrow_+$ to a_j precisely as to a_i. Thus $a_j, b_2, \ldots, b_k$ realizes π and thus $\pi \in \mathcal{A}_j$. The latter inclusion can be shown analogously. Recall that by Claim 2 (i) $|\mathcal{A}_i|$ and $|\mathcal{B}_i|$ are bounded exponentially. Thus when moving along $a_1, \ldots, a_k$ each of the components $\mathcal{A}_i$ and $\mathcal{B}_i$ can change at most exponentially many times, and in consequence, k is bounded exponentially. Finally, noting that the number of 1-types is also bounded exponentially we get the desired bound. ◀

The bounds in the both parts of the above corollary are essentially optimal. Exponentially long paths over $\{\downarrow_+\}$ can be easily enforced even in FO^2 by organizing, by means of unary predicates $P_0, \ldots, P_{n-1}$, a binary counter counting from 0 to $2^n - 1$ and requiring each node storing a value smaller than $2^n - 1$ to have a descendant storing the value greater by one. Let us see that the presence of $\downarrow$ allows to simply enforce doubly-exponential paths. We use unary predicates $N, P, P_0, \ldots, P_{n-1}, Q$. See Fig. 1. The intended long path is the path of elements in N. Every element in N is going to have 2^n children marked by P, each of which has a *local position* in the range $[0, 2^n - 1]$ encoded by means of $P_0, \ldots, P_{n-1}$. Reading the truth-values of Q as binary digits we can assume that the collection of the P-children of a node in N encodes its *global position* in the tree in the range $[0, 2^{2^n} - 1]$ (the i-th bit of this global position is 1 iff at the element at local position i the value of Q is true). It is then possible to say that each node in n whose global position is smaller than $2^{2^n} - 1$ has a child in N with the global position greater by 1. We skip here the details.

3.2 Bounded degree

Our next aim is to show that also the degree of nodes can be bounded.

▶ **Lemma 7.**

(i) *Let φ be a normal form $\mathrm{F}_1[\downarrow, \downarrow_+, \rightarrow, \rightarrow^+]$ formula. Let $\mathfrak{T} \models \varphi$. Then there exists a tree $\mathfrak{T}^* \models \varphi$ obtained by removing some subtrees from $\mathfrak{T}$ (and appropriately repairing the sibling relations), in which the degree of every node is bounded doubly exponentially in $\|\varphi\|$ by a fixed function $\mathfrak{f}'_d$.*

(ii) *Let φ be a normal form $\mathrm{F}_1[\downarrow]$, $\mathrm{F}_1[\downarrow_+]$ or $\mathrm{F}_1[\downarrow, \downarrow_+]$. Let $\mathfrak{T} \models \varphi$. Then there exists a tree $\mathfrak{T}^* \models \varphi$ obtained by removing some subtrees from $\mathfrak{T}$, in which the degree of every node is bounded exponentially in $\|\varphi\|$ by a fixed function $\mathfrak{f}'_s$.*

Proof. Let n be the width of φ.

(i) Consider any node $a \in T$. Let $a_1, \ldots, a_k$ be all the the children of a, listed from left to right. If for some $i < j$ the horizontal n-profiles of a_i and a_j are equal (note that $i > 1$ in this case) then we remove all the subtrees rooted at $a_i, \ldots, a_{j-1}$ and join a_{i-1} with a_j by $\rightarrow$. By arguments similar to those from the proof of Lemma 5 we can show that the profiles of the surviving elements of T do not change. Repeating this process as long as possible we eventually obtain a tree in which the number of children of a is bounded doubly exponentially (by the number of distinct horizontal n-profiles). We then repeat the process successively for all the nodes of $\mathfrak{T}$.

(ii) The $\mathcal{B}$ components of profiles do not behave monotonically along horizontal paths (as they do along vertical paths), thus we cannot use horizontal k-profiles to get the desired exponential bound on their length. We proceed in a slightly different manner. Consider any node $a \in T$. Let $\bar{a} = a_1, \ldots, a_k$ be the list of the children of a. For $1 \leq i \leq k$ let $types(a_i)$ be the set of types of size at most n realized in $\mathfrak{T}^{\downarrow}_{a_i}$ by tuples whose first element is a_i. For each type $\pi \in \bigcup_{i=1}^{k} types(a_i)$ mark n nodes in $\bar{a}$ such that $\pi \in types(a_i)$ (or all such nodes if there are less than n of them). This way we mark at most exponentially many children of a. Let us remove all the subtrees rooted at unmarked nodes from $\bar{a}$ and denote the obtained tree $\mathfrak{T}'$. We claim the the n-profiles of all the elements surviving the surgery do not change. Consider any $c \in T'$. Noting that the elements of $\mathfrak{T}'$ are related to each other by the navigational predicates $\downarrow_+, \downarrow$ exactly as they are related in $\mathfrak{T}$ we see that $\mathrm{prof}^{\mathfrak{T}'}_n(c).\mathcal{A} \subseteq \mathrm{prof}^{\mathfrak{T}}_n(c).\mathcal{A}$ and $\mathrm{prof}^{\mathfrak{T}'}_n(c).\mathcal{B} \subseteq \mathrm{prof}^{\mathfrak{T}}_n(c).\mathcal{B}$.

For $\supseteq$ we distinguish two case: the one in which c is in $\mathfrak{T}'^{\uparrow}_a$, and the other in which it is not. Let us sketch the arguments for the latter (the former is similar). Since c retains its subtree from $\mathfrak{T}$ it is clear that $\mathrm{prof}^{\mathfrak{T}}_n(c).\mathcal{B} \subseteq \mathrm{prof}^{\mathfrak{T}'}_n(c).\mathcal{B}$. To see that $\mathrm{prof}^{\mathfrak{T}}_n(c).\mathcal{A} \subseteq \mathrm{prof}^{\mathfrak{T}'}_n(c).\mathcal{A}$ take any $\pi \in \mathrm{prof}^{\mathfrak{T}}_n(c).\mathcal{A}$ and its any realization $c, b_1, \ldots, b_l \in T$. Split $b_1, \ldots, b_l$ into the disjoint tuples of nodes: let the first tuple $\bar{b}_0$ contain the nodes from $\mathfrak{T}^{\uparrow}_a$ and the other tuples $\bar{b}_1, \ldots, \bar{b}_s$ the nodes from the subtrees rooted at distinct nodes from $\bar{a}$ (note that $s < n$). The tuple $\bar{b}_0$ is retained in $\mathfrak{T}'$; the other tuples may be deleted, but due to our strategy of marking important nodes in $\bar{a}$ there exists a $1{-}1$ function returning for a tuple $\bar{b}_i$ a node in $\bar{a}$ surviving the surgery in whose subtree a tuple $\bar{b}'_i$ of type equal to the type of $\bar{b}_i$ exists. Using the elements $c, \bar{b}_0, \bar{b}'_1, \ldots, \bar{b}'_s$ we can now form a realization of π in $\mathfrak{T}'$ (starting from c). Thus $\pi \in \mathrm{prof}^{\mathfrak{T}'}_n(c).\mathcal{A}$, which finishes the argument.

Again, repeating the described process for all nodes we eventually obtain a tree $\mathfrak{T}^*$ in which the degree of every node is bounded exponentially and the n-profiles of all nodes remain as in $\mathfrak{T}$, and thus are φ-admissible. In effect $\mathfrak{T}^* \models \varphi$. ◀

The bounds on the degree of nodes in Lemma 7 are essentially optimal. In particular a doubly exponential chain of siblings can be enforced by means of $\rightarrow$ and $\downarrow$ (or $\rightarrow$ and $\downarrow_+$) similarly to a doubly exponential vertical root-to-leaf path: every element of the chain is required to have 2^n children storing the binary digits of a doubly exponential counter; the next sibling of a node a is forced to store the counter value greater by one than the value stored at a.

4 Complexity of satisfiability

In this section, using the results from Section 3 we establish the precise complexity of the satisfiability problem in all of the scenarios we consider.

4.1 Only descendant relation

Let us start with the observation that in the case when only the descendant relation is present in the navigational signature the complexity of F_1 is equal to the complexity of FO^2 and C^2.

▶ **Theorem 8.** *The satisfiability problem for* $\mathrm{F}_1[\downarrow_+]$ *is* ExpSpace*-complete.*

The lower bound is inherited from $\mathrm{FO}^2[\downarrow_+]$, [2], which in turn refers to ExpSpace-hardness of the so-called one-way two-variable guarded fragment, [9]. For the upper bound we design an alternating exponential time procedure. The result then follows from the well known fact that AExpTime=ExpSpace, [3]. The procedure first guesses the profile of the root and then guesses the profiles of its children, checking if the information recorded in the profiles is locally consistent, and if each guessed profile is φ-admissible. Further, it works in a loop, universally moving to one of the children, guessing profiles of its children and proceeding similarly.

Algorithm 1: Procedure F_1`[↓+]-sat-test`

Input: an $\mathrm{F}_1[\downarrow_+]$ normal form formula φ
- let n be the width of φ
- let $maxdepth := \mathfrak{f}_s(\|\varphi\|)$; let $maxdegree := \mathfrak{f}'_s(\|\varphi\|)$; % *cf. Cor. 6 and Lem. 7*
- let $level := 0$;
- **guess** an n-profile θ such that $\theta.\mathcal{A} = \{\alpha\}$; % *root*
- **while** $level < maxdepth$ **do**
  - if θ is not φ-admissible then **reject**
  - **guess** an integer $0 \leq k \leq maxdegree$; % *the number of children*
  - for $1 \leq i \leq k$ **guess** a profile θ_i;
  - if not *locally-consistent*$(\theta, \theta_1, \ldots, \theta_k)$ then **reject**;
  - if $\theta.\mathcal{B} = \{\alpha\}$ then **accept** % *a leaf reached; it must be* $k = 0$
  - $level := level + 1$;
  - **universally choose** $1 \leq i \leq k$; let $\theta := \theta_i$;
- **endwhile**
- **reject**
- **endprocedure**

The function *locally-consistent* checks whether, from a local point of view, a tree may have a node realizing an n-profile θ whose children realize n-profiles $\theta_1, \ldots, \theta_k$. It checks if $\theta.\mathcal{B}$ is the set of types which can be obtained, informally speaking, by putting $\theta.\alpha$ on top of a disjoint union of types taken from distinct $\theta_i.\mathcal{B}$ (possibly with their roots removed); and, analogously, verifies some natural conditions on the $\mathcal{A}$-components of θ and θ_i-s. We skip the details of this function.

▶ **Lemma 9.** *Procedure* F_1`[↓+]-sat-test` *accepts its input* φ *iff* φ *is satisfiable.*

Proof. Assume that φ is satisfiable. By Corollary 6 (ii) and Lemma 7 (ii) there exists a small model $\mathfrak{T} \models \varphi$. The procedure accepts φ by making all its guesses in accordance to $\mathfrak{T}$, i.e., in the first step it sets θ to be equal to the n-profile of the root of $\mathfrak{T}$ and then in each step it sets θ_i to be the n-profile of the i-th child of the previously considered element.

In the opposite direction, from an accepting (tree-)run t of the procedure we can naturally construct a tree structure $\mathfrak{T}_t$, with 1-types of elements as guessed during the execution. Our procedure guesses actually not only 1-types but full n-profiles of nodes. The function *locally-consistent* guarantees that the n-profiles of nodes $\mathfrak{T}_t$ are indeed as guessed. To see this, we first prove by induction on the height of nodes that the $\mathcal{B}$ components of profiles are

as required: the base step for leaves holds due to the acceptance condition of the procedure; then we move towards the root using the conditions of the function *locally-consistent*. For the $\mathcal{A}$-components we proceed by induction on the depth of nodes: the base case holds for the root by the requirement on the first profile from the procedure; then we move down the tree using the conditions of the function *locally-consistent* and the already proved fact that the $\mathcal{B}$-component are as required.

Since the procedure checks if each of the guessed n-profiles is φ-admissible, then by Lemma 3 we have that $\mathfrak{T}_t \models \varphi$. ◀

To finish the proof of Thm. 8 it remains to note that the values of *maxdepth* and *maxdegree* ensure that the algorithm works indeed in (alternating) exponential time.

4.2 Descendant and child

We first note that when the child relation $\downarrow$ is available in the signature then, in contrast to the case of FO^2 and C^2 the satisfiability problem becomes 2-ExpTime-hard. This can be shown by a simple adaptation of the 2-ExpTime-hardness proof for the unary negation fragment, UNFO, [17], which works in particular for UNFO[$\downarrow$] over trees. Actually some two-dimensional formulas appear in that proof, but their usage is not crucial and can be easily avoided.

▶ **Theorem 10.** *The satisfiability problem for* $\mathrm{F}_1[\downarrow]$ *is 2-*ExpTime*-hard.*

A matching upper bound for $\mathrm{F}_1[\downarrow, \downarrow_+]$ is easy to obtain using the tools we have already developed.

▶ **Theorem 11.** *The satisfiability problem for* $\mathrm{F}_1[\downarrow, \downarrow_+]$ *is 2-*ExpTime*-complete.*

Proof. The lower bound follows from Thm. 10. For the upper bound we just modify the procedure F_1`[`$\downarrow_+$`]-sat-test` from Section 4.1 in a natural way. By Corollary 6 (i) and Lemma 7 (i) we know that satisfiable formulas have models with doubly exponentially bounded paths and exponentially bounded degree of nodes. Thus we just change the initial value of *maxdepth* to $\mathfrak{f}_d(\|\varphi\|)$ and adjust the function *locally-consistent* by taking into account the presence of $\downarrow$ relation. The obtained procedure works in alternating exponential space. Since AExpSpace = 2-ExpTime [3] we get the desired result. ◀

4.3 Ordered trees

We conclude this section observing that the satisfiability problem of F_1 over ordered trees remains in 2-ExpTime.

▶ **Theorem 12.** *The satisfiability problem for* $\mathrm{F}_1[\downarrow, \downarrow_+, \rightarrow, \rightarrow^+]$ *is 2-*ExpTime*-complete.*

Proof. The lower bound follows from Thm. 10. For the upper bound we need another modification of the procedure F_1`[`$\downarrow_+$`]-sat-test`. As we want to fit in alternating exponential space we cannot this time allow ourselves for guessing at once all the children of a node (as there may be doubly exponentially many of them). Instead we start by guessing the leftmost one, and then move to the right until we reach the rightmost one. During this horizontal walk, at each node a we actually make a universal choice between continuing the walk to the right and moving down to the first child of a. Further, instead of n-profiles of nodes we guess horizontal-n-profiles of the form $(\alpha, \mathcal{A}, \mathcal{B}, \mathcal{A}_L, \mathcal{A}_R)$. The function `locally-consistent` takes this into account; it can be naturally designed to ensure that the the nodes guessed in an

accepting tree-run of the procedure indeed have the declared horizontal profiles. Note in particular that knowing the components $\mathcal{A}_L$, $\mathcal{A}_R$ and $\mathcal{B}$ of the horizontal profile of any child of a one can compute the $\mathcal{B}$ component of the profile of a. We leave the technical details of the required adaptation for the full version of the paper. ◀

5 Expressivity

We show that the expressive power of $\mathrm{F}_1[\tau_{bin}]$ is equal to the expressive power of $\mathrm{C}^2[\tau_{bin}]$ in the case of ordered trees, $\tau_{bin} = \{\downarrow, \downarrow_+, \rightarrow, \rightarrow^+\}$, and in the case of unordered trees accessible by both the descendant and the child relations, $\tau_{bin} = \{\downarrow, \downarrow_+\}$. Translation from C^2 to F_1 is easy. Consider, e.g., a subformula of the form $\exists^{\geq k} y \psi(x, y)$ and note that it can be written as $\exists y_1, \ldots, y_k(\bigwedge_{i \neq j} y_i \neq y_j \wedge \bigwedge_i \psi(x, y_i))$. Regarding the opposite direction, in the case of the ordered trees the equivalence is not very surprising: the power of the full navigational signature allows scanning trees in an ordered way and express the existence of patterns described by F_1 subformulas by a reuse of two variables. Actually, one can even directly translate F_1 to FO^2 without counting in this case. The equivalence over unordered trees is slightly harder and less obvious. The proof of the following theorem will be given in the full version of this paper.

▶ **Theorem 13.** *For any* $\mathrm{F}_1[\downarrow, \downarrow_+]$ *(*$\mathrm{F}_1[\downarrow, \downarrow_+, \rightarrow, \rightarrow^+]$*) sentence* φ *there is a* $\mathrm{C}^2[\downarrow, \downarrow_+]$ *(*$\mathrm{C}^2[\downarrow, \downarrow_+, \rightarrow, \rightarrow^+]$*) sentence* φ^* *such that for any tree* $\mathfrak{T}$ *we have* $\mathfrak{T} \models \varphi$ *iff* $\mathfrak{T} \models \varphi^*$*, and vice versa.*

Let us also collect here the results comparing the expressive power over trees of F_1, C^2 and two other two-variable logics: the two-variable guarded fragment, GF^2, and plain FO^2. By $A \prec B$ we denote that A is strictly less expressive than B (on the level of sentences), and by $A \equiv B$ we denote that A and B have the same expressive power.

▶ **Corollary 14.**
(i) *Over* $\tau = \{\downarrow, \downarrow_+, \rightarrow, \rightarrow^+\}$ *we have* $\mathrm{GF}^2[\tau] \equiv \mathrm{FO}^2[\tau] \equiv \mathrm{C}^2[\tau] \equiv \mathrm{F}_1[\tau]$.
(ii) *If* $\tau = \{\downarrow, \downarrow_+\}$ *then* $\mathrm{GF}^2[\tau] \prec \mathrm{FO}^2[\tau] \prec \mathrm{C}^2[\tau] \equiv \mathrm{F}_1[\tau]$.

Regarding (i): the translation of FO^2 to GF^2 can be done similarly to the translation of FO^2 to a variant of Core XPath in [15], equivalence of C^2 and FO^2 is shown in [1] and of C^2 and F_1 in Thm. 13. Regarding (ii): Let us assume that the signature contains no unary predicates and for $i \in \mathbb{N}$ let $\mathfrak{T}_i$ denote the tree consisting just of a root and its i children. The C^2 formula $\exists x \exists^{\geq 3} y\ x \downarrow_+ y$ distinguishes $\mathfrak{T}_3$ and $\mathfrak{T}_2$, while the FO^2 formula $\exists xy(\neg x \downarrow_+ y \wedge \neg y \downarrow_+ x \wedge x \neq y)$ distinguishes $\mathfrak{T}_2$ and $\mathfrak{T}_1$. It is not difficult to see that FO^2 cannot distinguish between $\mathfrak{T}_3$ and $\mathfrak{T}_2$ (use a simple 2-pebble game argument, cf. [1]) and that GF^2 cannot distinguish between $\mathfrak{T}_2$ and $\mathfrak{T}_1$ (use bisimulations).

6 Conclusion

We established the computational complexity of F_1 over unordered trees, showing its ExpSpace-completeness over trees accessible only by the descendant relation and 2-ExpTime-completeness in the presence of the child relation. The 2-ExpTime-upper bound holds also for ordered trees equipped additionally with the next-sibling and following-sibling relations. Deriving the complexities for the remaining combinations of the considered navigational symbols (e.g., $\mathrm{F}_1[\downarrow_+, \rightarrow]$) is not difficult, and we will do it in the full version of this paper.

We also proved that under two of the considered navigational scenarios F_1 is expressively equivalent to C^2. Extending the expressive power comparison between F_1 and C^2 (and other logics) to the signatures containing just one of $\downarrow$, $\downarrow_+$ is left as a future work.

Another interesting open problem is to formally compare the succinctness of F_1 versus FO^2 and C^2 over trees (and over words).

We worked with unranked trees, but it is not difficult to adapt all the results for the case of ranked trees. In particular the counters in the 2-ExpTime-lower bound proof (Thm. 10) can be organized as binary subtrees instead of horizontal chains of siblings.

References

1 Bartosz Bednarczyk, Witold Charatonik, and Emanuel Kieronski. Extending two-variable logic on trees. In *Computer Science Logic*, pages 11:1–11:20, 2017.

2 Saguy Benaim, Michael Benedikt, Witold Charatonik, Emanuel Kieronski, Rastislav Lenhardt, Filip Mazowiecki, and James Worrell. Complexity of two-variable logic on finite trees. *ACM Trans. Comput. Log.*, 17(4):32:1–32:38, 2016.

3 Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981. doi:10.1145/322234.322243.

4 Witold Charatonik, Emanuel Kieronski, and Filip Mazowiecki. Satisfiability of the two-variable fragment of first-order logic over trees. *CoRR*, abs/1304.7204, 2013.

5 Witold Charatonik and Piotr Witkowski. Two-variable logic with counting and a linear order. In *Computer Science Logic*, pages 631–647, 2015.

6 Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite model theory*. Perspectives in Mathematical Logic. Springer, 1995.

7 Kousha Etessami, Moshe Y. Vardi, and Thomas Wilke. First-order logic with two variables and unary temporal logic. *Inf. Comput.*, 179(2):279–295, 2002. doi:10.1006/inco.2001.2953.

8 Lauri Hella and Antti Kuusisto. One-dimensional fragment of first-order logic. In *Advances in Modal Logic 10*, pages 274–293, 2014.

9 Emanuel Kieronski. On the complexity of the two-variable guarded fragment with transitive guards. *Inf. Comput.*, 204(11):1663–1703, 2006.

10 Emanuel Kieronski. One-dimensional logic over words. In *Computer Science Logic*, pages 38:1–38:15, 2016.

11 Emanuel Kieronski and Antti Kuusisto. Complexity and expressivity of uniform one-dimensional fragment with equality. In *Mathematical Foundations of Computer Science, Part I*, pages 365–376, 2014.

12 Emanuel Kieronski and Antti Kuusisto. Uniform one-dimensional fragments with one equivalence relation. In *Computer Science Logic*, pages 597–615, 2015.

13 Andreas Krebs, Kamal Lodaya, Paritosh Pandya, and Howard Straubing. Two-variable logic with a between predicate. In *Logic in Computer Science*, 2016.

14 Antti Kuusisto. On the uniform one-dimensional fragment. In *Proceedings of Description Logic Workshop*, 2016.

15 Maarten Marx. First order paths in ordered trees. In *International Conference on Database Theory*, pages 114–128, 2005.

16 Dana Scott. A decision method for validity of sentences in two variables. *Journal Symbolic Logic*, 27:477, 1962.

17 Luc Segoufin and Balder ten Cate. Unary negation. *Logical Methods in Computer Science*, 9(3), 2013.

18 Larry J. Stockmeyer. *The Complexity of Decision Problems in Automata Theory and Logic*. PhD thesis, MIT, Cambridge, Massachusetts, USA, 1974.