Ordering Constraints over Feature Trees Expressed in Second-order Monadic Logic

Martin Müller Joachim Niehren

Programming Systems Lab, Universität des Saarlandes
66041 Saarbrücken, Germany, {mmueller,niehren}@ps.uni-sb.de

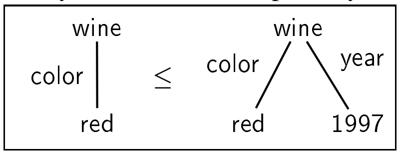
Abstract. The system $FT_\leq$ of ordering constraints over feature trees has been introduced as an extension of the system FT of equality constraints over feature trees. We investigate decidability and complexity questions for fragments of the first-order theory of $FT_\leq$. It is well-known that the first-order theory of FT is decidable and that several of its fragments can be decided in quasi-linear time, including the satisfiability problem of FT and its entailment problem with existential quantification $\varphi \models \exists x_1 \ldots \exists x_n \varphi'$. Much less is known on the first-order theory of $FT_\leq$. The satisfiability problem of $FT_\leq$ can be decided in cubic time, as well as its entailment problem without existential quantification. Our main result is that the entailment problem of $FT_\leq$ with existential quantifiers is decidable but PSPACE-hard. Our decidability proof is based on a new technique where feature constraints are expressed in second-order monadic logic with countably many successors SωS. We thereby reduce the entailment problem of $FT_\leq$ with existential quantification to Rabin's famous theorem on tree automata.

Keywords Feature logic, tree orderings, entailment, decidability, complexity, second-order monadic logic.

1 Introduction

Feature constraints have been used for describing records in constraint programming [1, 22, 21] and record like structures in computational linguistics [11, 10, 20, 16, 18]. Following [2, 4, 3], we consider feature constraints as predicate logic formulae interpreted in the structure of feature trees. We consider the system $FT_\leq$ of ordering constraints over feature trees [15, 12] which is as an extension of the system FT of equality constraints over feature trees. Ordering constraints in $FT_\leq$ are interpreted with respect to the weak subsumption ordering [7] on feature trees. Here, we investigate decidability and complexity questions for fragments of the first-order theory of $FT_\leq$.

A feature tree is a tree with unordered edges labeled by features and with possibly labeled nodes. Features are functional in that the features labeling the edges departing from the same node must be pairwise different. A feature tree τ_1 is smaller than a the feature tree τ_2 in the weak subsumption ordering if τ_1 has fewer edges and node labels than τ_2. In this case we write $\tau_1 \leq \tau_2$. An example is given in the picture.



The results in this paper hold for a countably infinite set of features f and a finite set of node labels a. We focus on the case of possibly infinite trees but we also consider the case of finite trees. The particular choice will be made explicit whenever necessary.

The constraints φ of $FT_{\leq}$ are defined by the following abstract syntax where x and x' are variables.

$$\varphi \ ::= \ x{\leq}x' \ \mid \ x[f]x' \ \mid \ a(x) \ \mid \ \varphi \wedge \varphi'$$

The semantics of $FT_{\leq}$ is given by the interpretation over feature trees where the symbol $\leq$ is interpreted as weak subsumption ordering. The constraints of FT have the same syntax as those of $FT_{\leq}$ but with equalities $x{=}y$ instead of ordering constraints $x{\leq}y$. Equalities are expressible in $FT_{\leq}$ since $x{=}y \leftrightarrow x{\leq}y \wedge y{\leq}x$ holds. The semantics of selection $x[f]y$ and labeling constraints $a(x)$ is the same in FT and in $FT_{\leq}$. For instance, both trees in the picture above are possible denotations for x in solutions of the constraint $\mathsf{wine}(x) \wedge x[\mathsf{color}]x' \wedge \mathsf{red}(x)$.

It is well-known that the first-order theory of FT is decidable [4] and that several of its fragments can be decided in quasi-linear time [1], including the satisfiability problem of FT and its entailment problem with existential quantification $\varphi \models \exists x_1 \ldots \exists x_n \varphi'$. Much less is known on the first-order theory of $FT_{\leq}$. The entailment problem $\varphi \models \varphi'$ of $FT_{\leq}$ has been shown to have cubic time complexity in [15]. It is however not known whether more expressive fragments of the first-order theory of $FT_{\leq}$ are decidable.

We consider the entailment problem $\varphi \models \exists x_1 \ldots \exists x_n \varphi'$of $FT_{\leq}$ with existential quantifiers. In the case of infinite trees, we show that this problem is at least PSPACE-hard. We prove this result by encoding the inclusion problem between regular word languages. Our proof makes essential use of infinite trees for encoding the Kleene star. When interpreted over the structure of finite trees, the entailment problem of $FT_{\leq}$ with existential quantifiers is at least coNP-hard. We prove this result by encoding the complement of the SAT-Problem of Boolean formulas. Here, we adapt a proof idea introduced by Henglein and Rehof [8].

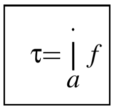
We prove that the entailment problem of $FT_{\leq}$ with existential quantifiers $\varphi \models \exists x_1 \ldots \exists x_n \varphi'$ is decidable, both in the case of finite trees and in the case of infinite trees. In the case of finite trees, we give a reduction to the weak second-order monadic logic WSωS with countably many successors [23] and in the case of infinite trees to the full second-order monadic logic SωS with countably many successors [17]. The idea to encode trees as sets of words is well-known, for instance from [5]. Feature constraints, however, have not yet been encoded in SωS. The reason is that it is impossible in SωS to express prefix closedness of tree domains and direct subtree relation $\tau[f]\tau'$ simultaneously. In this paper, we avoid the need to express prefix closedness by means of a semantics change (which we prove correct independently of our encoding).

Plan of the Paper. Section 2 introduces syntax and semantics of the constraint system $FT_{\leq}$. Section 3 illustrates the expressiveness of entailment with existential quantification and gives the lower bound complexity results. Section 4 defines second-order monadic logic and gives our reduction of entailment in $FT_{\leq}$ to validity in SωS resp. WSωS. Section 5 contains the correctness proof of our reduction. Section 6 summarizes. The full paper [14] extends the conference version with two appendices that contain all omitted proofs.

2 Syntax and Semantics of $\mathrm{FT}_{\leq}$

The constraint system $\mathrm{FT}_{\leq}$ is defined by a set of constraints together with an interpretation over feature trees. We assume an infinite set of *variables* ranged over by x, y, z, a countably infinite set $\mathcal{F}$ of *features* ranged over by f, g and a non-empty finite set $\mathcal{L}$ of *labels* ranged over by a, b.

Feature Trees. A *path* π is a word over features. The *empty path* is denoted by ε and the free-monoid concatenation of paths π and π' as $\pi\pi'$; we have $\varepsilon\pi = \pi\varepsilon = \pi$. Given paths π and π', π' is called a *prefix of* π if $\pi = \pi'\pi''$ for some path π''. A *tree domain* is a non-empty prefix closed set of paths. A *feature tree* τ is a pair (D, L) consisting of a tree domain D and partial labeling function $L : D \rightharpoonup \mathcal{L}$. Given a feature tree τ, we write D_τ for its tree domain and L_τ for its labeling function. A feature tree is called *finite* if its tree domain is finite, and *infinite* otherwise. A *node of* τ is an element of D_τ. A *leaf of* τ is a maximal node of τ. A node π of τ is *labeled with* a if $(\pi, a) \in L_\tau$. A node of τ is unlabeled if it is not labeled by any a. The *root* of τ is the node ε. For instance, $\tau = (\{\varepsilon, f\}, (f, a))$ is a finite tree that has a leaf f labeled with a and whose root is unlabeled.



Syntax and Semantics. An $\mathrm{FT}_{\leq}$ *constraint* φ is defined by the abstract syntax

$$\varphi \ ::= \ x{\leq}y \ \mid \ a(x) \ \mid \ x[f]y \ \mid \ \varphi_1 \wedge \varphi_2$$

An $\mathrm{FT}_{\leq}$ constraint is a conjunction of *basic constraints* which are either *inclusion constraints* $x{\leq}y$, *labeling constraints* $a(x)$, or *selection constraints* $x[f]y$.

We next define the structure $\mathrm{FT}_{\leq}$ over feature trees in which we interpret $\mathrm{FT}_{\leq}$ constraints. The signature of $\mathrm{FT}_{\leq}$ contains the binary relation symbols $\leq$, for every label a a unary relation symbol $a()$, and for every feature f a binary relation symbol $[f]$. In $\mathrm{FT}_{\leq}$ these relation symbols are interpreted such:

$$\begin{array}{rll} \tau_1{\leq}\tau_2 & \text{iff} & D_{\tau_1} \subseteq D_{\tau_2} \text{ and } L_{\tau_1} \subseteq L_{\tau_2} \\ \tau_1[f]\tau_2 & \text{iff} & D_{\tau_2} = \{p \mid f\pi \in D_{\tau_1}\} \text{ and } L_{\tau_2} = \{(\pi, a) \mid (f\pi, a) \in L_{\tau_1}\} \\ a(\tau) & \text{iff} & (\varepsilon, a) \in L_\tau \end{array}$$

First-Order Formulas. Let Φ and Φ' be first-order formulas built from $\mathrm{FT}_{\leq}$ constraints with the usual first order connectives. We call Φ *satisfiable* (valid) if Φ is satisfiable (valid) in the structure $\mathrm{FT}_{\leq}$. We say that Φ *entails* Φ', written $\Phi \models \Phi'$, if $\Phi \to \Phi'$ is valid, and that Φ is *equivalent* to Φ' if $\Phi_1 \leftrightarrow \Phi_2$ is valid. We denote with $V(\Phi)$ the set of variables occurring free in Φ and with $\mathcal{F}(\Phi)$ the set of features occurring in Φ.

We use the notation $x{\sim}a$ as an abbreviation for the formula $\exists y(x{\leq}y \wedge a(y))$. The formula $x{=}a$ means that x denotes the tree $(\{\varepsilon\}, \{(\varepsilon, a)\})$ and is defined as a short hand for the first-order formula $a(x) \wedge \forall y(a(y) \to x{\leq}y)$. We write $\bar{x}$ for a possibly empty word of variables $x_1 \ldots x_n$. In this case we also write $\exists\bar{x}\varphi$ instead of $\exists x_1 \ldots \exists x_n \varphi$.

As additional notation, we define extended first-order formulas for non-immediate subtree relations. Generalizing $[f]$, we introduce a new binary relation symbol $[\pi]$ for every path π. We also provide for extended formulas $x[\pi]y$ with the following semantics:

$$x[\varepsilon]y \leftrightarrow x{\leq}y \wedge y{\leq}x \qquad \text{and} \qquad x[\pi_1\pi_2]y \leftrightarrow \exists z(x[\pi_1]z \wedge z[\pi_2]y)$$

We will make use of the notations $x[\pi]y$, $x[\pi]_{\geq}y$, $x[\pi]_{\leq}y$, and $x[\pi]_{\geq}a$, which we consider as abbreviations for the following extended first order formulas:

$$\begin{aligned} x[\pi]_{\geq}y &\leftrightarrow \exists z(x[\pi]z \wedge y{\leq}z) \\ x[\pi]_{\leq}y &\leftrightarrow \exists z(x[\pi]z \wedge z{\leq}y) \\ x[\pi]_{\geq}a &\leftrightarrow \exists y(x[\pi]_{\geq}y \wedge a(y)) \end{aligned}$$

Alternative Definitions of Feature Trees. In the literature, there are two alternative definitions of feature trees [2, 3] distinct from ours. Accorsing to [2], every node must be labeled, and [3] requires exactly the leaves to be labeled. In contrast, we follow previous work of ours [15] and *allow* labels at all nodes but do *not require* any.

For equality constraints as in FT, the particular definition of feature trees does not matter. The reason is that the first-order theory of FT is completely axiomatizable [4] and that every definition of feature trees yields a model of the axiomatization of FT. All these models are distinct but their first-order theories coincide due to the complete axiomatization of FT.

With respect to ordering constraints as in $\mathrm{FT}_{\leq}$ the particular definition of feature trees does matter: Consider the existential formula Φ_1 ($a_1 \neq a_2$) and the first-order formula Φ_2:

$$\Phi_1 \;=\; x{\sim}a_1 \wedge x{\sim}a_2 \qquad\qquad \Phi_2 \;=\; \exists x \forall y\, x{\leq}y$$

The formula Φ_1 says that the label at the root of the denotation of x is compatible both with a_1 and a_2. Since $a_1 \neq a_2$, this is equivalent to saying that the root node of the denotation of x is unlabeled. Thus, Φ_1 is satisfiable in $\mathrm{FT}_{\leq}$, but not in a structure of feature trees where every node has to be labeled. The formula Φ_2 says that there exists a smallest feature tree with respect to the weak subsumption ordering. Such a tree exists in $\mathrm{FT}_{\leq}$, namely the tree $(\{\varepsilon\}, \emptyset)$. In contrast, there is no smallest tree in structures over feature trees that require all nodes or all leaves to be labeled. Thus, Φ_2 distinguishes the structure $\mathrm{FT}_{\leq}$ from those proposed for FT in [2, 3, 4].

3 Expressiveness of Entailment in $\mathrm{FT}_{\leq}$

We show the entailment problem of $\mathrm{FT}_{\leq}$ with existential quantification $\varphi \models \exists\bar{x}\varphi'$ to be surprisingly expressive. In the case of infinite trees, we obtain a stronger result than in the case of finite trees. Without existential quantifiers, the expressiveness of entailment is quite low, even in the case of infinite trees. This is shown in [15].

Theorem 1. *The satisfiability problem of* $\mathrm{FT}_{\leq}$ *can be tested in cubic time. The entailment problem* $\varphi \models \varphi'$ *of* $\mathrm{FT}_{\leq}$ *can be tested in cubic time.*

In [15] it is also shown that ordering constraints of $\mathrm{FT}_{\leq}$ (without existential quantification) have the independence property: If $\varphi \models \varphi_1 \vee \ldots \vee \varphi_n$ then there exists $1 \leq i \leq n$ such that $\varphi \models \varphi_i$. Independence fails in the presence of existential quantifiers since certain disjunctions can now be expressed by an entailment problem. For instance, if $a_1 \neq a_2$ then $x{\sim}a_1 \models x{\sim}a_2 \vee a_1(x)$ but neither $x{\sim}a_1 \models x{\sim}a_2$ nor $x{\sim}a_1 \models a_1(x)$.

3.1 Finite Trees

We show that the entailment problem of $\mathrm{FT}_{\leq}$ with existential quantifiers over finite trees is coNP-hard by reducing the complement of the propositional satisfiability problem SAT [6] to it Our encoding is based on an idea by Henglein's and Rehof's in [8]. They have considered entailment between ordering constraints over constructor trees (atomic subtyping) without existential quantification. The precise relationship between Henglein's and Rehof's result and ours is interesting and not obvious.

We assume an infinite set of boolean variables ranged over by u. A *clause* C is a finite disjunction of literals u or $\neg u$. We write *false* for the empty clause. A solution of a finite conjunction of clauses $\bigwedge_{i=1}^{n} C_i$ is a function β that assigns boolean values to boolean variables such that each of the C_i evaluates to *true* under β. The *clause satisfiability problem* (SAT) is whether a given conjunction $\bigwedge_{i=1}^{n} C_i$ has a solution. Without loss of generality we assume that no clause contains both a literal and its negation.

Proposition 2. *Fix k boolean variables $u_1, \ldots, u_k$ and a constraint variable x. There exists a mapping Ψ_x from clauses over $u_1, \ldots, u_k$ to existential* $\mathrm{FT}_{\leq}$ *formulas such that*

1. *for all clauses C (over $u_1, \ldots, u_k$) the size of $\Psi_x(C)$ is linear in k, and*
2. *for every conjunction of clauses $\bigwedge_{i=1}^{n} C_i$ (over $u_1, \ldots, u_k$) it holds that*

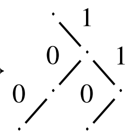
$$\bigwedge_{i=1}^{n} \Psi_x(C_i) \models \Psi_x(\mathit{false}) \quad \text{if} \quad \bigwedge_{i=1}^{n} C_i \text{ is unsatisfiable.}$$

The proof is given in Appendix A of the full paper. Here we illustrate the main ideas by an example. Consider the clauses $C_1 = \neg u_1 \vee u_3$, $C_2 = \neg u_1 \vee \neg u_3$, and $C_3 = u_1$ over three propositional variables u_1, u_2, and u_3, and observe that $C_1 \wedge C_2$ is satisfiable, while $C_1 \wedge C_2 \wedge C_3$ is non-satisfiable. Fix a variable x. Proposition 2 claims the existence of formulas $\Psi(C_1,x)$ through $\Psi(C_3,x)$ and $\Psi(\mathit{false},x)$ such that

$$\begin{array}{lcl} \Psi_x(C_1) \wedge \Psi_x(C_2) & \not\models & \Psi_x(\mathit{false}) \\ \Psi_x(C_1) \wedge \Psi_x(C_2) \wedge \Psi_x(C_3) & \models & \Psi_x(\mathit{false}) \end{array}$$

The formula $\Psi_x(C_1)$ and its minimal solution for x are depicted below. By $\Psi_x(C_1)$, the denotation of x is forced to have at least the paths in the tree on the right. These paths exactly correspond to the boolean valuations of u_1 through u_3 under which C_1 evaluates to false [8] (the features 1 and 0 correspond to the truth values *true* and *false*, resp.). While the trees of depth 3 may have exponentially many paths, we use the ordering to express sharing (*i.e.*, common lower bounds of different subtrees) and thus retain linear (space) complexity. Similarly, the constraint $\Psi_x(\mathit{false})$ forces the denotation of x to have *all* paths in $\{0,1\}^3$, and a conjunction $\bigwedge_{i=1}^{n} C_i$ is non-satisfiable if $\bigwedge_{i=1}^{n} \Psi_x(C_i)$ entails that x has all paths in $\{0,1\}^3$.

$$\begin{array}{l} \exists y \\ \exists y_0 \\ \exists y_1 \\ \exists z \end{array} \left(\begin{array}{l} x[1]y \\ y[0]y_0 \wedge z \leq y_0 \\ y[1]y_1 \wedge z \leq y_1 \\ z[0]y_0 \end{array} \right) \qquad x \mapsto$$



Corollary 3. *Over finite trees, the entailment problem of* $\mathrm{FT}_{\leq}$ *with existential quantification is co*NP*-hard.*

3.2 Infinite Trees

We show how to linearly reduce the inclusion problem between regular languages over finite words to the entailment problem $\varphi \models \exists\bar{x}\varphi'$ over $\mathrm{FT}_{\leq}$. Since this problem is well-known to be PSPACE-complete [9, 19], we obtain PSPACE-hardness of entailment.

Theorem 4. *Over infinite trees, the entailment problem of* $\mathrm{FT}_{\leq}$ *with existential quantification* $\varphi \models \exists\bar{x}\varphi'$ *is PSPACE-hard.*

Proof. Follows from Proposition 5 below. □

The idea of the proof of Theorem 4 is to encode regular sets of words (over features) as feature trees. For instance, the set $\{1,111\}$ can be described by the feature tree τ with

$$D_\tau = \{1,11,111\} \quad \text{and} \quad L_\tau = \{(1,a),(111,a)\}$$

in that $\{1,111\}$ is equal to the set of nodes of τ that are labeled with a. We consider regular expressions over a finite subset F of $\mathcal{F}$ defined as usual.

$$S \ ::= \ \emptyset \mid \varepsilon \mid f \mid S^* \mid S_1 \cup S_2 \mid S_1 S_2 \qquad \text{where } f \in F$$

Each regular expression S defines a set $\mathcal{L}(S)$ of finite words over F. Without loss of generality, we assume that every non-trivial regular expression $S \neq \emptyset$ does not contain the symbol $\emptyset$ at all. Hence, we can assume $\mathcal{L}(S) \neq \emptyset$ if $S \neq \emptyset$.

Proposition 5. *Let x be an arbitrary variable. For every pair of regular expressions S_1 and S_2 there exist existential formulas $\Theta(x,S_1)$ and $\Theta(x,S_2)$ with size linear in the size of S_1 and S_2, respectively, such that*

$$\Theta(x,S_1) \models \Theta(x,S_2) \quad \textit{if and only if} \quad \mathcal{L}(S_2) \subseteq \mathcal{L}(S_1).$$

The proof is given below after the necessary definitions and two auxiliary Lemmas. We define the formula $\Theta(x,S)$ inductively over the form of S.

$$\begin{array}{lcl}
\Theta(x,S) & = & \exists y \exists z (y \leq x \wedge \Theta'(y,S,z) \wedge a(z)) \\
\quad \Theta'(x,\emptyset,y) & = & true \\
\quad \Theta'(x,\varepsilon,y) & = & y \leq x \\
\quad \Theta'(x,f,y) & = & \exists z (x[f]z \wedge y \leq z) \\
\quad \Theta'(x,S_1 \cup S_2,y) & = & \Theta'(x,S_1,y) \wedge \Theta'(x,S_2,y) \\
\quad \Theta'(x,S^*,y) & = & \exists z \exists z' (y \leq z \wedge \Theta'(z,S,z') \wedge z' = z \wedge z \leq x) \\
\quad \Theta'(x,S_1 S_2,y) & = & \exists z (\Theta'(x,S_1,z) \wedge \Theta'(z,S_2,y))
\end{array}$$

Lemma 6. *The formula $\Theta(x,S)$ has size linear in the size of S.*

Lemma 7. *Let $x \neq y$, α be a variable assignment, and S be a regular expression.*

1. *α is a solution of $\Theta'(x,S,y)$ if and only if $\forall \pi \in \mathcal{L}(S) : \alpha(x)[\pi]_{\geq}\alpha(y)$.*
2. *α is a solution of $\Theta(x,S)$ if and only if $\forall \pi \in \mathcal{L}(S) : (\pi, a) \in L_{\alpha(x)}$.*

Proof. Structural induction over S. For the proof see Appendix A of the full paper. $\square$

Proof of Proposition 5. Assume that $\mathcal{L}(S_2) \not\subseteq \mathcal{L}(S_1)$. Then there exists $\pi \in \mathcal{L}(S_2)$ such that $\pi \notin \mathcal{L}(S_1)$. By Lemma 26 there exists a solution α of $\Theta(x,S_1)$ such that $(\pi,a) \notin L_{\alpha(x)}$. By Lemma 7, α is not a solution of $\Theta(x,S_1)$. Hence $\Theta(x,S_1) \not\models \Theta(x,S_2)$.

For the converse, assume $\mathcal{L}(S_2) \subseteq \mathcal{L}(S_1)$. Then apparently for all α the following property holds: $\forall\pi \in \mathcal{L}(S_1) : (\pi,a) \in L_{\alpha(x)}$ implies $\forall\pi \in \mathcal{L}(S_2) : (\pi,a) \in L_{\alpha(x)}$. By Lemma 7, this is equivalent to saying that every a solution of $\Theta(x,S_1)$ is a solution of $\Theta(x,S_2)$, *i.e.*, $\Theta(x,S_1) \models \Theta(x,S_2)$. $\square$

4 Deciding Entailment with Existential Quantifiers

In this Section we prove the decidability of the entailment problem of $\mathrm{FT}_{\leq}$ by reduction to Rabin's decidability result for second-order monadic logic. In this proof, our particular choice of a definition of feature trees will turn out crucial.

Second-Order Monadic Logic (SωS and WSωS). We recall the definitions of *second-order monadic logic with countably many successors* SωS [17] and of *weak second-order monadic logic with countably many successors* WSωS [23]. Syntactically, SωS and WSωS coincide. We assume an additional infinite set of *path variables* denoted by p that is disjoint from the variables denoted by x. Formulas ψ of SωS and WSωS are built from variables x and p and features f.

$$
\begin{aligned}
w &::= p \mid \varepsilon \mid fw \\
\psi &::= w{\in}x \mid w{=}w' \mid \psi \wedge \psi' \mid \neg\psi \mid \forall p\psi \mid \forall x\psi
\end{aligned}
$$

The semantics of SωS is defined as follows. A path variable p is interpreted as a path (a word over features) and a variable x as a set of words over features. The denotation of ε is the empty path and the denotation of fw is the path obtained by concatenation f in front of the denotation of w. The membership constraint $w{\in}x$ holds if the denotation of w is a member of the denotation of x. The equality constraint $w{=}w'$ holds if the denotations of w and w' are equal. The semantics of WSωS coincides with the semantics of SωS except that in WSωS a variable x denotes a *finite* set of paths.

As derived forms we will use the following formulas with their usual semantics:

$$\neg\psi,\ \exists p\psi,\ \exists x\psi,\ \psi \rightarrow \psi'\ \psi \leftrightarrow \psi'$$

Theorem 8 (Rabin,Thatcher,Wright [17, 23]). *The satisfiability problems of* WSωS *and* SωS *are decidable.*

Theorem 9. *The entailment problem of* $\mathrm{FT}_{\leq}$ *with existential quantification* $\varphi \models \exists\bar{x}\varphi'$ *is decidable, both when interpreted over finite feature trees or over infinite feature trees.*

The proof is developed in this section and given at its end. The details of the proof cover the rest of the paper. The underlying idea is quite simple. If we ignore labels then a feature tree coincides with its domain which is a set of paths. Therefore, ordering constraints over trees can be translated into monadic second order logic. This idea is well known for constructor trees [5]. The pitfall here is that only prefix closed sets of paths correspond to a feature tree. Prefix-closedness can be expressed in SωS but not simultaneously with the direct subtree relation $\tau[f]\tau'$. The reason is that SωS allows for concatenation to the left πf but not for concatenation to the right $f\pi$ (or vice versa).

We avoid the need to express prefix closedness by first changing semantics (Section 4.1). We define the structure $FT^{-}_{\leq}$ of sufficiently labeled feature trees and reduce the entailment problem of $FT_{\leq}$ to the entailment problem of $FT^{-}_{\leq}$. In a second step (Section 4.2), we encode entailment relative to $FT^{-}_{\leq}$ into formulas of second order monadic logic with countably many successors (WSωS for finite trees and SωS for infinite trees).

4.1 Changing Semantics

Definition 10. We call a feature tree τ *sufficiently labeled* if for every $\pi \in D_\tau$ there exists a path π' and a label a such $(\pi\pi', a) \in L_\tau$.

Note that a finite feature tree is sufficiently labeled if and only if all its leaves are labeled. Every sufficiently labeled feature tree can be identified with a unique n-tuple of non-empty sets of paths, and vice versa, if n is the number of labels in $\mathcal{L}$. For every label a we define a function γ_a from feature trees to non-empty sets of paths:

$$\gamma_a(\tau) = \{\pi \mid L_\tau(\pi) = a\}$$

For $\mathcal{L}=\{a_1, \ldots, a_n\}$ we define $\gamma(\tau)$ as the following n-tuple of sets of paths:

$$\gamma(\tau) = (\gamma_{a_1}, \ldots, \gamma_{a_n})$$

Proposition 11. *The mapping γ from sufficiently labeled feature trees to n-tuples of pairwise disjoint sets of words with non-empty union is one-to-one and onto. Furthermore, τ is a finite tree if and only if every component of $\gamma(\tau)$ is finite.*

The proof is given in Appendix B.1 of the full paper. Note that we need not require prefix closedness for the sets in the domain of γ, since the domain of a sufficiently labeled feature tree τ is uniquely determined by its labeling function L_τ. This observation is crucial for our reduction to second-order monadic logic. Note also that the notion of sufficient labeling does not make sense for the alternative notions of feature trees mentioned above [2, 3].

Definition 12. The structure $FT^{-}_{\leq}$ is the restriction of the structure $FT_{\leq}$ to the domain of sufficiently labeled feature trees.

We may interpret $FT^{-}_{\leq}$ either over finite trees or over possible infinite trees. Whenever this choice matters, we will made it explicit.

The first-order theories of $FT_\leq$ and $FT^-_\leq$ differ. For instance, consider the following existential formula Φ_3 (or alternatively the formula Φ_2 from above):

$$\Phi_3 = \exists x(x \leq x_1 \wedge x \leq x_2)$$

Φ_3 requires for all τ_1 and τ_2 that there exists τ such that $\tau \leq \tau_1$ and $\tau \leq \tau_2$. Φ_3 is valid over $FT_\leq$ but not valid over $FT^-_\leq$. In $FT_\leq$ one may always chose $\tau = (\varepsilon, \emptyset)$. This is impossible in $FT^-_\leq$ since $(\varepsilon, \emptyset)$ is not sufficiently labeled. Even worse, if $\tau_1 = (\{\varepsilon\}, \{(\varepsilon, a_1)\})$, $\tau_2 = (\{\varepsilon\}, \{(\varepsilon, a_2)\})$, and $a_1 \neq a_2$ then we cannot find any appropriate tree τ in $FT^-_\leq$.

We need distinct notations for entailment with respect to $FT_\leq$ and $FT^-_\leq$. For this purpose, we write $\Phi \models_{FT_\leq} \Phi'$ and $\Phi \models_{FT^-_\leq} \Phi'$. The next proposition claims that, in some sense, this distinction is not necessary for the formulas of interest.

We fix a label $b \in \mathcal{L}$ throughout the rest of the paper. Given some feature $g \in \mathcal{F}$ we define a function η_g that maps a constraint φ to a first-order formula over constraints as follows:

$$\eta_g(\varphi) = \varphi \wedge \bigwedge_{y \in \mathcal{V}(\varphi)} \exists y' (y[g]y' \wedge y'=b)$$

The use of the feature b in the definition of $\eta_g(\varphi)$ is left implicit in our notation. This is because it does not really matter (and could even be circumvented technically).

Proposition 13. *Let φ and φ' be constraints such that $\mathcal{V}(\varphi') \subseteq \mathcal{V}(\varphi)$, $\bar{x}$ a sequence of variables, and g a feature. If $g \notin \mathcal{F}(\varphi \wedge \varphi')$ then $\varphi \models_{FT_\leq} \exists \bar{x} \varphi'$ is equivalent to $\eta_g(\varphi) \models_{FT^-_\leq} \exists \bar{x} \varphi'$, both over finite trees and over infinite trees.*

Proof. This proof of this proposition is technically involved. It is given in Section 5, the two implications being subject of Propositions 20 and 24. □

Note that Proposition 13 fails when $\eta_g(\varphi)$ is replaced by φ. This can also illustrated by formula Φ_3. As argued above, Φ_3 is valid over $FT_\leq$ but not over $FT^-_\leq$. In order to relate this fact to Proposition 13, let φ_3 be the tautological constraint $x_1 \leq x_1 \wedge x_2 \leq x_2$ such that $\mathcal{V}(\varphi_3) \subseteq \mathcal{V}(\varphi_3)$. Now, $\varphi_3 \models_{FT_\leq} \Phi_3$ but $\varphi_3 \not\models_{FT^-_\leq} \Phi_3$. However, $\eta_g(\varphi_3) \models_{FT^-_\leq} \Phi_3$ where $\eta_g(\varphi_3) = \exists x_1' (x_1[g]x_1' \wedge x_1'=b) \wedge \exists x_2' (x_2[g]x_2' \wedge x_2'=b)$.

4.2 Reduction to SωS or WSωS

We next define a mapping from first-order formulas over ordering constraints (interpreted over $FT^-_\leq$) to formulas of second-order monadic logic with countably many successors. We will make use of the following abbreviations:

$$x \cap y = \emptyset = \neg \exists p (p \in x \wedge p \in y)$$

For every variable x and label a let x_a be a fresh variable. Suppose that $\mathcal{L} = \{a_1, \ldots, a_n\}$. Here comes the definition of the mapping $[\![\ \_\ ]\!]$:

$$
\begin{array}{lcl}
[\![a(x)]\!] & = & \varepsilon{\in}x_a \\
[\![x[f]y]\!] & = & \bigwedge_{i=1}^{n} \forall p\,(fp{\in}x_{a_i} \leftrightarrow p{\in}y_{a_i}) \\
[\![x{\leq}y]\!] & = & \bigwedge_{i=1}^{n} x_{a_i}{\subseteq}y_{a_i} \\
[\![\varphi\wedge\varphi']\!] & = & [\![\varphi]\!]\wedge[\![\varphi']\!] \\
[\![\neg\varphi]\!] & = & \neg\,[\![\varphi]\!] \\
[\![\exists x\varphi]\!] & = & \exists x_{a_1}\ldots\exists x_{a_n}\,((\bigwedge_{\substack{i,j=1\\ i\neq j}}^{n} x_{a_i}{\cap}x_{a_j}{=}\emptyset\,)\wedge\exists p\,(\bigvee_{i=1}^{n} p{\in}x_{a_i})\ \wedge [\![\varphi]\!])
\end{array}
$$

Proposition 14. *A first-order formula Φ is valid over* $\mathrm{FT}_{\leq}^{-}$ *interpreted over finite (resp. infinite) trees if and only if its translation* $[\![\varphi]\!]$ *is valid over* WSωS *(resp.* SωS*).*

Proof. If α is a solution of Φ then α' with $\alpha'(x_a) = \gamma_a(\alpha(x))$ is a solution of $[\![\Phi]\!]$. If β is a solution of $[\![\Phi]\!]$ then the mapping β' with $\beta'(x) = \gamma^{-1}(\beta(x_{a_1}),\ldots,\beta(x_{a_1}))$ is a solution of Φ. The existence of the inverse mapping γ^{-1} of γ is proved by Proposition 11.

Proof of Theorem 9. We wish to decide an entailment problem of the form $\varphi \models_{\mathrm{FT}_{\leq}} \exists\bar{x}\varphi'$. We choose a feature g not occurring in φ or φ' (this exists since the set of all features $\mathcal{F}$ is infinite). By Proposition 13 it is sufficient to decide the entailment propositions $\eta_g(\varphi) \models_{\mathrm{FT}_{\leq}^{-}} \exists\bar{x}\varphi'$ over $\mathrm{FT}_{\leq}^{-}$. By Proposition 14, $\eta_g(\varphi) \models_{\mathrm{FT}_{\leq}^{-}} \exists\bar{x}\varphi'$ holds if and only if the translation $[\![\eta_g(\varphi) \rightarrow \exists\bar{x}\varphi']\!]$ is a valid formula of WSωS in the case of finite trees and of SωS in the case of infinite trees. The validity of these formulas is decidable by Rabin's Theorem 8. □

5 Changing Semantics is Correct

We prove that the semantics change from $\mathrm{FT}_{\leq}$ to $\mathrm{FT}_{\leq}^{-}$ is correct in the sense of Proposition 13. All omitted proofs can be found in Appendix B of the full paper.

5.1 Entailment in $\mathrm{FT}_{\leq}^{-}$ Implies Entailment in $\mathrm{FT}_{\leq}$

Throughout this Section we are interested in entailment propositions $\varphi \models \exists\bar{x}\varphi'$ where $g \notin \mathcal{F}(\varphi\wedge\varphi')$ for a fixed feature g.

Adding Labels. We define a mapping δ_g from feature trees to feature trees. Intuitively, $\delta_g(\tau)$ is obtained by adding a leaf $(\pi g, b)$ to every node π of τ. Formally, we assume a feature tree τ such that $\pi g \notin D_\tau$ for all π.

$$
\begin{aligned}
D_{\delta_g(\tau)} &= D_\tau \cup \{\pi g \mid \pi \in D_\tau\} \\
L_{\delta_g(\tau)} &= L_\tau \cup \{(\pi g, b) \mid \pi \in D_\tau\}
\end{aligned}
$$

Lemma 15. *If τ is a feature tree such that $g \notin \mathcal{F}(D_\tau)$ for all π then $\delta_g(\tau)$ is a feature tree that is sufficiently labeled.*

Proof. The assumption $\pi g \notin D_\tau$ for all π implies that $L_{\delta_g(\tau)}$ is a partial function such that $\delta_g(\tau)$ is indeed a feature tree. $\square$

Lemma 16. *Assume $g \notin \mathcal{F}(\alpha(x))$ for x and $g \notin \mathcal{F}(\varphi)$. If α is a solution of φ in $\mathrm{FT}_\leq$ then $\delta_g \circ \alpha$ is a solution of φ in $\mathrm{FT}^-_\leq$.*

Deleting Labels. There exists a left-inverse δ_g^{-1} of the function δ_g on all feature trees τ such that $g \notin \mathcal{F}(\tau)$. For arbitrary τ, we define a feature tree $\delta_g^{-1}(\tau)$ as follows:

$$\begin{aligned} D_{\delta_g^{-1}(\tau)} &= D_\tau \setminus \{\pi g \pi' \mid \pi, \pi' \in \mathcal{F}^*\} \\ L_{\delta_g^{-1}(\tau)} &= L_\tau \setminus \{(\pi, a) \mid \pi = \pi' g \pi'',\ a \in \mathcal{L}\} \end{aligned}$$

Lemma 17. *If $g \notin \mathcal{F}(\tau)$ then $\delta_g^{-1}(\delta_g(\tau)) = \tau$.*

Lemma 18. *Let $g \notin \mathcal{F}(\varphi)$. If α is a solution of φ in $\mathrm{FT}^-_\leq$ then $\delta_g^{-1} \circ \alpha$ is a solution of φ in $\mathrm{FT}_\leq$.*

Lemma 19 Correctness. *Let φ be a constraint with $g \notin \mathcal{F}(\varphi)$, $\bar{x}$ a sequence of variables, and α a variable assignment such that $g \notin \mathcal{F}(\alpha(y))$ for all $y \in \mathcal{V}(\varphi \wedge \exists \bar{x} \varphi')$. Then α is a solution of $\exists \bar{x} \varphi$ over $\mathrm{FT}_\leq$ if and only if $\delta_g \circ \alpha$ is a solution of $\exists \bar{x} \varphi$ over $\mathrm{FT}^-_\leq$.*

Proof. Let α be a solution of $\exists \bar{x} \varphi$ over $\mathrm{FT}_\leq$. There exists a sequence of trees $\bar{\tau}$ such that $\alpha[\bar{\tau}/\bar{x}]$ is a solution of φ over $\mathrm{FT}_\leq$. Since $g \notin \mathcal{F}(\varphi)$, the mapping $\delta_g^{-1} \circ (\alpha[\bar{\tau}/\bar{x}])$ is also a solution of φ by Lemma 18. The latter variable assignment coincides with $\alpha[\delta_g^{-1}(\bar{\tau})/\bar{x}]$ since we have assumed $g \notin \mathcal{F}(\alpha(y))$ for all y. Thus $\delta_g \circ (\alpha[\delta_g^{-1}(\bar{\tau})/\bar{x}])$ is a solution of φ over $\mathrm{FT}^-_\leq$ (Lemma 16), which implies that $\delta_g \circ \alpha$ is a solution of $\exists \bar{x} \varphi$ over $\mathrm{FT}^-_\leq$.

For the converse, assume that $\delta_g \circ \alpha$ is a solution of $\exists \bar{x} \varphi$ over $\mathrm{FT}^-_\leq$. There exists a sequence of trees $\bar{\tau}$ such that $(\delta_g \circ \alpha)[\bar{\tau}/\bar{x}]$ is a solution of φ over $\mathrm{FT}^-_\leq$. Hence, $\delta_g^{-1} \circ ((\delta_g \circ \alpha)[\bar{\tau}/\bar{x}])$ is a solution of φ over $\mathrm{FT}_\leq$(Lemma 18). Also, $\delta_g^{-1} \circ \delta_g \circ \alpha = \alpha$ due of Lemma 17 and $g \notin \alpha(y)$ for all y. Thus:

$$\delta_g^{-1} \circ ((\delta_g \circ \alpha)[\bar{\tau}/\bar{x}]) = (\delta_g^{-1} \circ \delta_g \circ \alpha)[\delta_g^{-1}(\bar{\tau})/\bar{x}] = \alpha[\delta_g^{-1}(\bar{\tau})/\bar{x}]$$

This proves that α is a solution of $\exists \bar{x} \varphi$ over $\mathrm{FT}_\leq$. $\square$

Proposition 20. *Let $g \notin \mathcal{F}(\varphi \wedge \varphi')$ and $\mathcal{V}(\exists \bar{x} \varphi') \subseteq \mathcal{V}(\varphi)$. If $\eta_g(\varphi) \models_{\mathrm{FT}^-_\leq} \exists \bar{x} \varphi'$ then $\varphi \models_{\mathrm{FT}_\leq} \exists \bar{x} \varphi'$.*

Proof. Let $g \notin \mathcal{F}(\varphi \wedge \varphi')$, $\mathcal{V}(\exists \bar{x} \varphi') \subseteq \mathcal{V}(\varphi)$, and $\eta_g(\varphi) \models_{\mathrm{FT}^-_\leq} \exists \bar{x} \varphi'$. We have to show that every solution α of φ in $\mathrm{FT}_\leq$ is also a solution of $\exists \bar{x} \varphi'$. Since the number of features in $\mathcal{F}$ is infinite, we only need to consider solutions α of φ such that $g \notin \mathcal{F}(\alpha(x))$ for all $x \in \mathcal{V}(\varphi)$. The proof of this fact is delegated to Lemma 29 in Appendix B.2.

Since α is a solution of φ, and $g \notin \mathcal{F}(\alpha(x))$ for all $x \in \mathcal{V}(\varphi \wedge \exists \bar{x} \varphi')$, $\delta_g \circ \alpha$ is a solution of φ (Lemma 19). From the definition of δ_g it follows that $\delta_g \circ \alpha$ is also a solution of

$\bigwedge_{y\in\mathcal{V}(\varphi)} \exists y' (y[g]y' \wedge y'=b)$, *i.e.*, $\delta_g \circ \alpha$ is a solution of $\eta_g(\varphi)$ over $\mathrm{FT}^-_{\leq}$. Entailment as assumed implies that $\delta_g \circ \alpha$ is a solution of $\exists\bar{x}\varphi'$ over $\mathrm{FT}^-_{\leq}$. Thus α is also a solution of $\exists\bar{x}\varphi'$ over $\mathrm{FT}_{\leq}$ (Lemma 19). □

5.2 Least Solutions

We will exploit the completeness of the satisfiability test for ordering constraints over feature trees given in [15]. This test also computes the least solution of a satisfiable constraint. The form of these least solutions can be interpreted as a criterion for entailment. We call a constraint φ F1F2-closed if it satisfies the following properties:

F1.1	$x{\leq}x$ *in* φ	if $x \in \mathcal{V}(\varphi)$
F1.2	$x{\leq}z$ *in* φ	if $x{\leq}y$ *in* φ and $y{\leq}z$ *in* φ
F2	$x'{\leq}y'$ *in* φ	if $x[f]x'$ *in* φ, $x{\leq}y$ *in* φ and $y[f]y'$ *in* φ

Lemma 21. *There exists a cubic time algorithm that given a constraint φ either detects the unsatisfiability of φ, or proves its satisfiability and returns an F1F2-closed constraint equivalent to φ.*

Proof. We first consider an extended set of rules that defines a satisfiability test. Let *false* and $x{\sim}y$ be auxiliary constraints where *false* denotes inconsistency and the semantics of $x{\sim}y$ is given by the equivalence $x_1{\sim}x_2 \leftrightarrow \exists x(x_1{\leq}x \wedge x_2{\leq}x)$. We call a constraint φ F-*closed* if it satisfies all properties F1–F5 where:

F3.1	$x{\sim}y$ *in* φ	if $x{\leq}y$ *in* φ
F3.2	$x{\sim}z$ *in* φ	if $x{\leq}y$ *in* φ and $y{\sim}z$ *in* φ
F3.3	$x{\sim}y$ *in* φ	if $y{\sim}x$
F4	$x'{\sim}y'$ *in* φ	if $x[f]x'$ *in* φ, $x{\sim}y$ *in* φ and $y[f]y'$ *in* φ
F5	*false in* φ	if $x(a) \wedge x{\sim}y \wedge y(a')$ *in* φ and $a \neq a'$

For every constraint φ an equivalent F-closed constraint φ' can be computed in cubic time applying the rules in F exhaustively. Furthermore, φ is satisfiable if and only if φ' does not contain *false*. Deleting the auxiliary constraints $x_1{\sim}x_2$ from φ' transforms φ' into an equivalent F1F2-closed constraint in linear time. □

We will use a syntactic description of the least solution of a satisfiable constraint in order to derive properties of entailment in $\mathrm{FT}_{\leq}$(see [15]). We define *syntactic entailment* judgements of the form $\varphi \vdash x[\pi]_{\geq}y$ and $\varphi \vdash x[\pi]_{\geq}a$ as follows.

$$\begin{array}{rl}
\varphi \vdash x[\varepsilon]_{\geq}y & \text{if } y{\leq}x \textit{ in } \varphi \\
\varphi \vdash x[f]_{\geq}y & \text{if } x[f]y \textit{ in } \varphi \\
\varphi \vdash x[\pi_1\pi_2]_{\geq}y & \text{if exists } z \text{ such that } \varphi \vdash x[\pi_1]_{\geq}z \text{ and } \varphi \vdash z[\pi_2]_{\geq}y \\
\varphi \vdash x[\pi]_{\geq}a & \text{if exists } z \text{ such that } \varphi \vdash x[\pi]_{\geq}z \text{ and } a(z) \textit{ in } \varphi
\end{array}$$

Proposition 22 Least Solutions. *Let φ be satisfiable and F1F2-closed. For every variable $x \in \mathcal{V}(\varphi)$, and all a,π,z the following two equivalences hold:*

$$\begin{array}{rcl}
\varphi \models_{\mathrm{FT}_{\leq}} \exists z\, x[\pi]_{\geq}z & \textit{iff} & \textit{exists } z' \textit{ such that } \varphi \vdash x[\pi]_{\geq}z' \\
\varphi \models_{\mathrm{FT}_{\leq}} x[\pi]_{\geq}a & \textit{iff} & \varphi \vdash x[\pi]_{\geq}a
\end{array}$$

Proof. The implications from the right to the left hold because syntactic entailment is correct with respect to semantic entailment in $\mathrm{FT}_{\leq}$. For the converse implication, we define the least solution least_φ of φ such that for all $x \in \mathcal{V}(\varphi)$.

$$\begin{aligned} D_{\mathsf{least}_\varphi(x)} &= \{\pi \mid \text{exists } z \text{ such that } \varphi \vdash x[\pi]_{\geq} z\} \\ L_{\mathsf{least}_\varphi(x)} &= \{(\pi, a) \mid \varphi \vdash x[\pi]_{\geq} a\} \end{aligned}$$

Without loss of generality, we can assume that φ is F-complete. First note that completion with $\mathsf{F1}$–$\mathsf{F5}$ does never derive *false* since φ is satisfiable. Second, since φ is $\mathsf{F1F2}$-closed, completion may only add auxiliary constraints $x{\sim}y$, which does not affect the validity of judgements $\varphi \vdash x[\pi]_{\geq} z$ and $\varphi \vdash x[\pi]_{\geq} a$. In [15] it is proved that least_φ is a solution of φ whenever φ is $\mathsf{F1}$-$\mathsf{F5}$-closed.

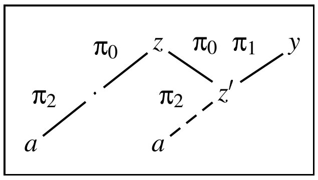
Since syntactic entailment is correct with respect to semantic entailment in $\mathrm{FT}_{\leq}$ it is clear the least_φ is smaller than every solution of φ, *i.e.*, least_φ is the least solution of φ: If $\varphi \not\vdash x[\pi]_{\geq} z'$ for all z' then $\pi \notin D_{\mathsf{least}_\varphi(x)}$, *i.e.*, $\varphi \not\models_{\mathrm{FT}_{\leq}} \exists z x[\pi]_{\geq} z$. In analogy, if $\varphi \not\vdash x[\pi]_{\geq} a$ then $(\pi, a) \notin L_{\mathsf{least}_\varphi(x)}$, *i.e.*, $\varphi \not\models_{\mathrm{FT}_{\leq}} x[\pi]_{\geq} a$. $\square$

5.3 Entailment in $\mathrm{FT}_{\leq}$ Implies Entailment in $\mathrm{FT}_{\leq}^{-}$

For every constraint φ let $-\varphi$ be the constraint that is obtained from φ by inverting all its ordering constraints, *i.e.*, by replacing $x{\leq}y$ with $y{\leq}x$. We define:

$$\varphi \vdash x[\pi]_{\leq} y \quad \text{iff} \quad -\varphi \vdash x[\pi]_{\geq} y$$

Lemma 23 Mountain Chains. *Let α be a solution of φ and assume variables y, z, z', paths π_0, π_1, π_2 and a label a. If $\varphi \models \exists z' (z[\pi_0]_{\leq} z' \wedge y[\pi_1]_{\geq} z')$ and $(\pi_0\pi_2, a) \in L_{\alpha(z)}$ then $(\pi_1\pi_2, a) \in L_{\alpha(y)}$.*



Proof. By induction on the length of π_0 we can show that $(\pi_2, a) \in L_{\alpha(z')}$ such that $(\pi_1\pi_2, a) \in L_{\alpha(y)}$. $\square$

Proposition 24. *Suppose $\mathcal{V}(\exists\bar{x}\varphi') \subseteq \mathcal{V}(\varphi)$ and $g \notin \mathcal{F}(\varphi \wedge \varphi')$. If $\varphi \models_{\mathrm{FT}_{\leq}} \exists\bar{x}\varphi'$ then $\eta_g(\varphi) \models_{\mathrm{FT}_{\leq}^{-}} \exists\bar{x}\varphi'$.*

Proof. Suppose $\mathcal{V}(\varphi) \cap \mathcal{V}(\bar{x}) = \emptyset$ and let α be a solution of $\eta_g(\varphi)$ over $\mathrm{FT}_{\leq}^{-}$. Note that $\mathcal{V}(\varphi') \subseteq \mathcal{V}(\bar{x}) \cup \mathcal{V}(\varphi)$. We have to construct a solution α' of $\varphi \wedge \varphi'$ over $\mathrm{FT}_{\leq}^{-}$ which coincides with α on the variables in $\mathcal{V}(\varphi)$.

We define $\alpha'(y)$ for all $y \in \mathcal{V}(\bar{x})$. Since $\alpha'(y)$ must be sufficiently labeled, it suffices to define its labeling function. Suppose φ and φ' to be $\mathsf{F1F2}$ closed. For $y \in \mathcal{V}(\bar{x})$ we define

L1	$(\pi, a) \in L_{\alpha'(y)}$	if	$\varphi' \vdash y[\pi]_{\geq} a$
L2	$(\pi g, b) \in L_{\alpha'(y)}$	if	exists z such that $\varphi' \vdash y[\pi]_{\geq} z$
L3	$(\pi_1\pi_2, a) \in L_{\alpha'(y)}$	if	$\begin{cases} \text{exists } z \in \mathcal{V}(\varphi) \text{ and } z' \text{ such that} \\ \varphi' \vdash y[\pi_1]_{\geq} z',\ \varphi' \vdash z[\pi_0]_{\leq} z' \text{ and } (\pi_0\pi_2, a) \in L_{\alpha(z)} \end{cases}$

(Compare condition L3 with the mountain situation depicted above.) For all $y \in \mathcal{V}(\bar{x})$, we define $D_{\alpha'(y)} = \{\pi \mid \pi \text{ is a prefix of } \pi' \text{ and } (\pi', a) \in L_{\alpha(y)}\}$. For all $y \notin \mathcal{V}(\bar{x})$ we set $\alpha'(y) = \alpha(y)$. It is clear that $\alpha'(y)$ is sufficiently labeled because of L2. It is also clear that α' is a solution of φ since α' coincides with α on $\mathcal{V}(\varphi)$. It remains to show that α' is a solution of φ', *i.e.*, that α' satisfies all basic constraints in φ'. Here, we only consider a single case. The complete case distinction is three pages long and given in the proof of Lemma 30 in Appendix B.4. Consider the case $x[f]y$ *in* φ', $x \in \mathcal{V}(\varphi)$, $y \in \mathcal{V}(\bar{x})$, and $(\pi, a) \in L_{\alpha'(y)}$ because of L2 or L3. We show $(f\pi, a) \in L_{\alpha'(x)}$.

L2 exists π' and z such that $\varphi' \vdash y[\pi']_{\geq} z$, $\pi = \pi' g$ and $a = b$. Since $x[f]y \in \varphi'$, $\varphi' \vdash x[f\pi']_{\geq} z$ such that $\varphi \models_{\mathrm{FT}_{\leq}} \exists z (x[f\pi']_{\geq} z)$ since $x \in \mathcal{V}(\varphi)$. Proposition 22 implies the existence of $z' \in \mathcal{V}(\varphi)$ such that $\varphi \vdash x[f\pi]_{\geq} z'$ and thus $\eta_g(\varphi) \vdash x[f\pi g]_{\geq} b$. Since α is a solution of $\eta_g(\varphi)$, we have $(f\pi' g, b) \in L_{\alpha(x)}$, *i.e.*, $(f\pi, a) \in L_{\alpha'(x)}$.

L3 Let $(\pi, a) \in L_{\alpha'(y)}$ since there exists $z \in \mathcal{V}(\varphi)$, z', π_0, π_1, and π_2 with $\varphi' \vdash y[\pi_1]_{\geq} z'$, $\varphi' \vdash z[\pi_0]_{\leq} z'$, $(\pi_0\pi_2, a) \in L_{\alpha(z)}$, and $\pi = \pi_1\pi_2$. Since $x[f]y$ we also have $\varphi' \vdash x[f\pi_1]_{\geq} z'$. Since $x, z \in \mathcal{V}(\varphi)$ this implies $\varphi \models_{\mathrm{FT}_{\leq}} \exists z' (z[\pi_0]_{\leq} z' \wedge x[f\pi_1]_{\geq} z')$. Now, Lemma 23 and $(\pi_0\pi_2, a) \in L_{\alpha(z)}$ imply $(f\pi_1\pi_2, a) \in L_{\alpha(x)}$, *i.e.*, $(f\pi, a) \in L_{\alpha'(x)}$. $\square$

6 Conclusion and Future Work

We have investigated decidability and complexity questions for fragments of the first-order theory of ordering constraints over feature trees ($\mathrm{FT}_{\leq}$). We have proved that the entailment problem of $\mathrm{FT}_{\leq}$ with existential quantifiers is coNP hard over finite trees, PSPACE hard over infinite trees, and decidable in both cases. We have related $\mathrm{FT}_{\leq}$ to the monadic second-order logic with multiple successors. At least two questions on the first-order theory of $\mathrm{FT}_{\leq}$ have been left for further research. Is its full first-order theory of $\mathrm{FT}_{\leq}$ decidable? And, how does it relate to monadic second-order logic?

Acknowledgments. The research reported in this paper has been supported by the the Esprit Working Group CCL II (EP 22457), the SFB 378 at the Universität des Saarlandes. We thank Jean-Marc Talbot, Sophie Tison, and Marc Tommasi for inspiring discussions. We also thank Ralf Treinen for discussion of Henglein's and Rehof's result, for careful proof reading and invaluable remarks on drafts of this paper.

References

1. H. Aït-Kaci and A. Podelski. Towards a Meaning of Life. *The Journal of Logic Programming*, 16(3 and 4):195–234, July, Aug. 1993.
2. H. Aït-Kaci, A. Podelski, and G. Smolka. A feature-based constraint system for logic programming with entailment. *Theoretical Computer Science*, 122(1–2):263–283, Jan. 1994.
3. R. Backofen. A Complete Axiomatization of a Theory with Feature and Arity Constraints. *The Journal of Logic Programming*, 1995. Special Issue on Computational Linguistics and Logic Programming.
4. R. Backofen and G. Smolka. A Complete and Recursive Feature Theory. *Theoretical Computer Science*, 146(1–2):243–268, July 1995.

5. H. Comon. Sequentiality, second-order monadic logic and tree automata. In K. Dexter, editor, *Proceedings of the Logic in Computer Science Conference*, pages 508–517, 1995.
6. S. A. Cook. The Complexity of Theorem-Proving Procedures. In *Annual ACM Symposium on Theory of Computing*, pages 151–158, New York, 1971. ACM.
7. J. Dörre. Feature logics with weak subsumption constraints. In *Annual Meeting of the ACL (Association of Computational Logics)*, pages 256–263, 1991.
8. F. Henglein and J. Rehof. The Complexity of Subtype Entailment for Simple Types. In *12th IEEE Symposium on Logic in Computer Science*, Warsaw, Poland, 1997. to appear.
9. J. D. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, MA, 1979.
10. R. M. Kaplan and J. Bresnan. Lexical-Functional Grammar: A Formal System for Grammatical Representation. pages 173–381. The MIT Press, Cambridge, MA, 1982.
11. M. Kay. Functional Grammar. In C. Chiarello et al., editor, *Proc. of the 5th Annual Meeting of the Berkeley Linguistics Society*, pages 142–158, 1979.
12. M. Müller. Ordering Constraints over Feature Trees with Ordered Sorts. In P. Lopez, S. Manandhar, and W. Nutt, editors, *Computational Logic and Natural Language Understanding*, Lecture Notes in Artificial Intelligence, to appear, 1997.
13. M. Müller and J. Niehren. Entailment for Set Constraints is not Feasible. Technical report, Programming Systems Lab, Universität des Saarlandes, 1997. Available at `http://www.ps.uni-sb.de/~mmueller/papers/conp97.html`.
14. M. Müller and J. Niehren. Ordering Constraints over Feature Trees Expressed in Second-order Monadic Logic. Technical report, Programming Systems Lab, Universität des Saarlandes, 1997. Technical Report.
15. M. Müller, J. Niehren, and A. Podelski. Ordering Constraints over Feature Trees. In G. Smolka, editor, *3rd International Conference on Principles and Practice of Constraint Programming*, 1997.
16. C. Pollard and I. Sag. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. Cambridge University Press, Cambridge, England, 1994.
17. M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.
18. W. C. Rounds. Feature Logics. In J. v. Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*. Elsevier Science Publishers B.V. (North Holland), 1997.
19. H. Seidl. Deciding Equivalence of Finite Tree Automata. *SIAM Journal of Computing*, 19(3):424–437, June 1990.
20. S. Shieber. *An Introduction to Unification-based Approaches to Grammar*. CSLI Lecture Notes No. 4. Center for the Study of Language and Information, 1986.
21. G. Smolka. The Oz Programming Model. In J. van Leeuwen, editor, *Computer Science Today*, Lecture Notes in Computer Science, vol. 1000, pages 324–343. Springer-Verlag, Berlin, Germany, 1995.
22. G. Smolka and R. Treinen. Records for Logic Programming. *The Journal of Logic Programming*, 18(3):229–258, Apr. 1994.
23. J. W. Thatcher and J. B. Wright. Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2(1):57–81, August 1967. Published by Springer-Verlag NY Inc.

A Proofs on Expressiveness

A.1 Finite Trees

Proposition 2. Fix k boolean variables $u_1,\ldots,u_k$ and a constraint variable x. There exists a mapping Ψ_x from clauses over $u_1,\ldots,u_k$ to existential $\mathrm{FT}_{\leq}$ formulas such that

1. For all clauses C (over $u_1,\ldots,u_k$) the size of $\Psi_x(C)$ is linear in k, and
2. for every conjunction of clauses $\bigwedge_{i=1}^{n} C_i$ (over $u_1,\ldots,u_k$) the following holds.

$$\bigwedge_{i=1}^{n} \Psi_x(C_i) \models \Psi_x(\mathit{false}) \quad \text{iff} \quad \bigwedge_{i=1}^{n} C_i \text{ is unsatisfiable.}$$

Proof. Assume that $\bigwedge_{i=1}^{n} C_i$ contains the variables $u_1,\ldots,u_k$. The idea of [8] is to represent every boolean variable assignment β on $\{u_1,\ldots,u_k\}$ as a path π_β:

$$\pi_\beta = \overline{\beta(u_k)}\ldots\overline{\beta(u_1)} \qquad \text{where} \quad \overline{\mathit{true}} = 1 \text{ and } \overline{\mathit{false}} = 0.$$

A set of boolean variable assignment B can be represented as feature that we define next. We say that

$$\beta \ \mathit{in}\ \tau \quad \text{if} \quad \pi_\beta \in D_\tau$$

and we encode a set B of boolean variable assignments as the smallest feature tree $\tau(B)$ with

$$\beta \in \tau(B) \quad \text{if} \quad \beta \in B\,.$$

Let τ^k be the complete unlabeled feature tree of depth k over features $\{0,1\}$. We now define the formulas $\Psi_x(C_i)$ such that

$$\alpha \text{ is a solution of } \bigwedge_{i=1}^{n} \Psi_x(C_i) \quad \text{iff} \quad \tau(\mathit{Sol}(\neg \bigwedge_{i=1}^{n} C_i)) \leq \alpha(x) \tag{1}$$

$$\alpha \text{ is a solution of } \Psi_x(\mathit{false},x) \quad \text{iff} \quad \tau^k \leq \alpha(x) \tag{2}$$

where $\mathit{Sol}(\neg\bigwedge_{i=1}^{n} C_i)) = \{\beta \mid \beta \text{ is a solution of } \neg\bigwedge_{i=1}^{n} C_i)\})$. Before doing so, we verify that the above two properties justify Proposition 2:

$$\begin{array}{lll}
\bigwedge_{i=1}^{n} \Psi_x(C_i) \models \Psi_x(\mathit{false},x) & \text{iff} & \forall\alpha, \alpha \models \bigwedge_{i=1}^{n} \Psi_x(C_i) : \alpha \models \Psi_x(\mathit{false},x) \\
& \text{iff} & \forall\alpha : \alpha \models \bigwedge_{i=1}^{n} \Psi_x(C_i) \ \rightarrow\ \tau^k \leq \alpha(x) \\
& \text{iff} & \tau^k \leq \tau(\mathit{Sol}(\neg\bigwedge_{i=1}^{n} C_i))) \qquad (*) \\
& \text{iff} & \bigwedge_{i=1}^{n} C_i \text{ is unsatisfiable}
\end{array}$$

The step marked $(*)$ exploits that, by (1), $\tau(\mathit{Sol}(\neg\bigwedge_{i=1}^{n} C_i))$ is a solution of $\bigwedge_{i=1}^{n} \Psi_x(C_i)$. We define the existential formulae $\Psi_x(C)$ as follows. For every $1 \leq i \leq k$ and every clause C over variables in $\{u_1,\ldots u_k\}$ let

$$\delta_i(C) = 1 \text{ if } \neg u_i \text{ in } C, \qquad \delta_i(C) = 0 \text{ if } u_i \text{ in } C, \qquad \delta_i(C) = 2 \text{ otherwise.}$$

Without loss of generality we assume that there exists no clause C and boolean variable u_i such that both u_i in C and $\neg u_i$ in C. Every clause C corresponds to a path $p(C)$ given by $p(C) = \delta_k(C)\ldots\delta_1(C)$. The formula $\Psi_x(C)$ is defined by recursion over $p(C)$.

$$\begin{array}{lcl}
\Psi_x(\varepsilon, y) & = & a(y) \\
\Psi_x(1p, y) & = & \exists y' (y[1]y' \wedge \Psi_x(p, y')) \\
\Psi_x(0p, y) & = & \exists y' (y[0]y' \wedge \Psi_x(p, y')) \\
\Psi_x(2p, y) & = & \exists y_0 \exists y_1 \exists y_2 (y[1]y_1 \wedge y[0]y_0 \wedge y_2 \leq y_1 \wedge y_2 \leq y_0 \wedge \Psi_x(p, y_2))
\end{array}$$

Since every step of this definition introduces at most three new variables, these formulae have size linear in the number of variables k and clauses n. It is easy to verify that every solution α of $\Psi_x(\mathit{false}, x)$ satisfies (2), *i.e.*, $\tau^k \leq \alpha(x)$. The fact that $\bigwedge_{i=1}^n \Psi_x(C_i)$ satisfies (1) is proved by induction over the number of variables. For more details see the report [13]. ◻

A.2 Infinite Trees

Proposition 5. Let x be an arbitrary variable. For every pair of regular expressions S_1 and S_2 there exist existential formulas $\Theta(x, S_1)$ and $\Theta(x, S_2)$ with size linear in the size of S_1 and S_2, respectively, such that

$$\Phi(x, S_1) \models \Phi(x, S_2) \text{ if and only if } \mathcal{L}(S_2) \subseteq \mathcal{L}(S_1).$$

Proof. $\Rightarrow$**:** Assume that $\mathcal{L}(S_2) \not\subseteq \mathcal{L}(S_1)$. Then there exists $\pi \in \mathcal{L}(S_2)$ such that $\pi \notin \mathcal{L}(S_1)$. By Lemma 26 there exists an α such that $\alpha \models \Theta(x, S_1)$ but $(\pi, a) \notin L_{\alpha(x)}$. By Lemma 7.2, $\alpha \not\models \Theta(x, S_1)$. Hence $\Theta(x, S_1) \not\models \Theta(x, S_2)$.

$\Leftarrow$**:** Assume $\mathcal{L}(S_2) \subseteq \mathcal{L}(S_1)$. Then apparently for all α

$$(\forall \pi \in \mathcal{L}(S_1) : (\pi, a) \in L_{\alpha(x)}) \text{ implies } (\forall \pi \in \mathcal{L}(S_2) : (\pi, a) \in L_{\alpha(x)})$$

holds. By Lemma 7.2, this is equivalent to saying that for all α: $\alpha \models \Theta(x, S_1)$ implies $\alpha \models \Theta(x, S_2)$, *i.e.*, $\Theta(x, S_1) \models \Theta(x, S_2)$. ◻

Lemma 25. *Let S be a regular expression and τ, τ' trees. If $\forall \pi \in \mathcal{L}(S^*) : \tau'[\pi]_{\geq} \tau$, then there exists τ'' such that $\tau \leq \tau'' \leq \tau'$ and $\tau''[\pi]_{\geq} \tau''$ for all $\pi \in \mathcal{L}(S^*)$.*

Proof. We define τ'' as the greatest tree which is smaller than the subtrees of τ' at π for all $\pi \in \mathcal{L}(S^*)$. This is well-defined as we can set

$$D_{\tau''} = \bigcap \{ D_{\tau'_\pi} \mid \tau'[\pi]_{\geq} \tau'_\pi,\ \pi \in \mathcal{L}(S^*) \} \qquad L_{\tau''} = \bigcap \{ L_{\tau'_\pi} \mid \tau'[\pi]_{\geq} \tau'_\pi,\ \pi \in \mathcal{L}(S^*) \}$$

where the intersection of partial functions is defined as the partial function corresponding to the intersection of their graphs.

It is clear that $\tau \leq \tau''$ (since $\forall \pi \in \mathcal{L}(S^*) : \tau'[\pi]_{\geq} \tau$ and τ'' is the greatest tree with this property), and $\tau'' \leq \tau'$ (since $\mathcal{L}(S) \neq \emptyset$ and hence $\varepsilon \in \mathcal{L}(S^*)$). Now pick an arbitrary $\pi \in \mathcal{L}(S^*)$. It is clear that $\pi \in D_{\tau''}$. Hence there exists τ''_π such that $\tau''[\pi]_{\geq} \tau''_\pi$. We have to show that $\tau''[\pi]_{\geq} \tau''$, *i.e.*, $D_{\tau''} \subseteq D_{\tau''_\pi}$ and $L_{\tau''} \subseteq L_{\tau''_\pi}$.

$D_{\tau''} \subseteq D_{\tau''_\pi}$**:** Let $\pi_1 \in D_{\tau''}$ and assume $\pi_1 \notin D_{\tau''_\pi}$. By definition of τ'' we know that

- $\forall \pi_2 \in \mathcal{L}(S^*)\ \exists \tau'_{\pi_2} : \tau'[\pi_2]_\geq \tau'_{\pi_2} \wedge \pi_1 \in D_{\tau'_{\pi_2}}$
- $\exists \pi_3 \in \mathcal{L}(S^*)\ \exists \tau'_{\pi\pi_3} : \tau'[\pi\pi_3]_\geq \tau'_{\pi\pi_3} \wedge \pi_1 \notin D_{\tau'_{\pi\pi_3}}$

This is a contradiction since for arbitrary π_3 it holds that $\pi\pi_3 \in \mathcal{L}(S^*)$.

$L_{\tau''} \subseteq L_{\tau''_\pi}$**:** Similar. □

Lemma 7.

1. Suppose $x \neq y$. Then $\alpha \models \Theta'(x,S,y)$ if and only if $\forall \pi \in \mathcal{L}(S) : \alpha(x)[\pi]_\geq \alpha(y)$.
2. $\alpha \models \Theta(x,S)$ if and only if $\forall \pi \in \mathcal{L}(S) : (\pi, a) \in L_{\alpha(x)}$.

Proof. 1. By structural induction over S. Assume $x \neq y$.

- $\alpha \models \Theta'(x,\varepsilon,y)$ iff $\alpha \models y \leq x$ iff $\alpha(y) \leq \alpha(x).\varepsilon = \alpha(x)$.
- $\alpha \models \Theta'(x,\emptyset,y)$ iff $\alpha \models \mathit{true}$ iff $\alpha(x)[\pi]_\geq \alpha(y)$ for all $\pi \in \mathcal{L}(\emptyset) = \emptyset$.
- $\alpha \models \Theta'(x,a,y)$ iff $\alpha \models \exists z (x[a]z \wedge y \leq z)$ iff $\alpha(y) \leq \alpha(x).a$
- $\alpha \models \Theta'(x,S^*,y)$ iff $\alpha \models \exists z \exists z' (y \leq z \wedge \Theta'(z,S,z') \wedge z{=}z' \wedge z \leq x)$ iff

$$\exists \tau_z \forall \pi \in \mathcal{L}(S) :\ \alpha(y) \leq \tau_z \wedge \tau_z[\pi]_\geq \tau_z \wedge \tau_z \leq x$$

  This immediately yields

$$\exists \tau_z \forall \pi \in \mathcal{L}(S^*) :\ \alpha(y) \leq \tau_z \wedge \tau_z[\pi]_\geq \tau_z \wedge \tau_z \leq x$$

  and this implies $\forall \pi \in \mathcal{L}(S^*) : \alpha(x)[\pi]_\geq \alpha(y)$.
  For the inverse direction assume $\forall \pi \in \mathcal{L}(S^*) : \alpha(x)[\pi]_\geq \alpha(y)$. By Lemma 25 there exists a feature tree τ such that

$$(i)\ \alpha(y) \leq \tau \leq \alpha(x) \qquad \text{and} \qquad (ii)\ \forall \pi \in \mathcal{L}(S^*) : \tau[\pi]_\geq \tau.$$

  Let $\alpha' = \alpha, z \mapsto \tau, z' \mapsto \tau$. Then, by (i), $\alpha' \models y \leq z \wedge z = z' \wedge z \leq x$, and, by (ii) and induction assumption, $\alpha' \models \Theta(z,S,z')$. In combination this is $\alpha \models \Theta(x,S^*,y)$.
- $\alpha \models \Theta'(x,S_1 \cup S_2,y)$ iff $\alpha \models \Theta'(x,S_1,y) \wedge \Theta'(x,S_2,y)$ iff $\forall \pi \in \mathcal{L}(S_1) : \alpha(x)[\pi]_\geq \alpha(y), \forall \pi' \in \mathcal{L}(S_2) : \alpha(x)[\pi']_\geq \alpha(y)$ iff $\forall \pi \in \mathcal{L}(S_1 \cup S_2) : \alpha(x)[\pi]_\geq \alpha(y)$.
- $\alpha \models \Theta'(x,S_1S_2,y)$ iff $\alpha \models \exists z(\Theta'(x,S_1,z) \wedge \Theta'(z,S_2,y))$ iff $\exists \tau : \alpha, z \mapsto \tau \models \Theta'(x,S_1,z) \wedge \Theta'(z,S_2,y)$. By induction assumption this is equivalent to

$$\exists \tau \forall \pi \in \mathcal{L}(S_1) \forall \pi' \in \mathcal{L}(S_2) : \alpha(x)[\pi]_\geq \tau \text{ and } \tau[\pi']_\geq \alpha(y)$$

  But since $\mathcal{L}(S_1) \neq \emptyset$ and $\mathcal{L}(S_2) \neq \emptyset$ this in turn is equivalent to

$$\forall \pi_1 \in \mathcal{L}(S_1) \forall \pi_2 \in \mathcal{L}(S_2) : \alpha(x)[\pi_1\pi_2]_\geq \alpha(y) \ \Leftrightarrow\ \forall \pi \in \mathcal{L}(S_1S_2) : \alpha(x)[\pi]_\geq \alpha(y)\,.$$

  In this equivalence, the implication from left to right is again easy. For the inverse assume $\alpha(x)[\pi_1\pi_2]_\geq \alpha(y)$ for all $\pi_1 \in \mathcal{L}(S_1), \pi_2 \in \mathcal{L}(S_2)$, define τ by

$$D_\tau = \hat{S}_2 \qquad L_\tau = \{(\pi\pi', b) \mid (\pi', b) \in L_{\alpha(y)}, \pi \in \mathcal{L}(S_2)\}$$

  and note that τ, by construction, satisfies $\tau[\pi_2]_\geq \alpha(y)$ and $\alpha(x)[\pi_1]_\geq \tau$ for all $\pi_1 \in \mathcal{L}(S_1)$ and $\pi_2 \in \mathcal{L}(S_2)$.

2. $\Rightarrow$: Assume $\alpha \models \Theta(x,S)$. Then exist τ_y and τ_z such that $\alpha, y \mapsto \tau_y, z \mapsto \tau_z \models y \leq x \wedge \Theta'(y,S,z) \wedge a(z)$. By case (1) of this Lemma this implies for all $\pi \in \mathcal{L}(S)$ that $\tau_y[\pi]_\geq \tau_z$ and $\tau_y \leq \alpha(x)$. Hence, for all $\pi \in \mathcal{L}(S)$, $(\pi, a) \in L_{\alpha(x)}$.

   $\Leftarrow$: Assume that for all $\pi \in \mathcal{L}(S) : (\pi, a) \in L_{\alpha(x)}$ and define $\beta = \alpha, y \mapsto \alpha(x), z \mapsto (\varepsilon, \{(\varepsilon, a)\})$. Then β trivially satisfies $y \leq x \wedge a(z)$, and furthermore for all $\pi \in \mathcal{L}(S) : \beta(y)[\pi]_\geq \leq \beta(z)$. Hence, by case (1) of this Lemma, $\beta \models y \leq x \wedge \Theta'(y,S,z) \wedge a(z)$ and thus $\alpha \models \Theta(x,S)$. □

Lemma 26. *For all x and S there exists* α *such that* $\{\pi \mid (\pi, a) \in L_{\alpha(x)} = \mathcal{L}(S)$.

Proof. The valuation α, which maps x to τ where D_τ is the smallest prefixed-closed domain containing $\mathcal{L}(S)$ and $(\pi, a) \in L_\tau$ iff $\pi \in \mathcal{L}(S)$, is a well-formed $FT_\leq$ valuation and satisfies the right hand side of Lemma 7.2.

B Proofs on Semantics Change

B.1 Suffiently Labeled Feature Trees

Proposition 11 The mapping γ from sufficiently labeled feature trees to n-tuples of pairwise disjoint sets of words with non-empty union is one-to-one and onto. Furthermore, τ is a finite tree if and only if every component of $\gamma(\tau)$ is finite.

Proof. Let τ be a sufficiently labeled feature tree. Since $\varepsilon \in D_\tau$ there exists a path π and a label a such that $(\varepsilon\pi, a) \in L_\tau$. Hence $\cup_{i=1}^n \gamma_{a_i}(\tau)$ is nonempty. The sets $\gamma_{a_i}(\tau)$ are pairwise disjoint since L_τ has to be a partial function. It is also clear that $\cup_{i=1}^n \gamma_{a_i}(\tau)$ is finite if τ is finite. The converse follows from the fact that a sufficiently labeled infinite tree has infinitely many labeled nodes.

In order to prove that γ is one-to-one and onto, we define the inverse mapping of γ as follows. Let $(\Pi_1, \ldots, \Pi_n)$ be pairwise disjoint sets of words over features that have a nonempty union. We define $\gamma^{-1}(\Pi_1, \ldots, \Pi_n)$ as follows:

$$\begin{aligned} D_{\gamma^{-1}(\Pi_1,\ldots,\Pi_n)} &= \{\pi \mid \pi \text{ is a prefix of some word in } \cup_{i=1}^n \Pi_i\} \\ L_{\gamma^{-1}(\Pi_1,\ldots,\Pi_n)} &= \{(\pi, a_i) \mid 1 \leq i \leq n,\ \pi \in \Pi_i\} \end{aligned}$$

Since $\cup_{i=1}^n \Pi_i$ is assumed non-empty, we have $\varepsilon \in D_{\gamma^{-1}(\Pi_1,\ldots,\Pi_n)}$, which is also prefix closed by construction. The relation $L_{\gamma^{-1}(\Pi_1,\ldots,\Pi_n)}$ is a partial function since all Π_i are assumed pairwise disjoint. Hence $\gamma^{-1}(\Pi_1, \ldots, \Pi_n)$ is a feature tree, which clearly is sufficiently labeled.

It is quite obvious that γ^{-1} is in fact the inverse function of γ, *i.e.*, that $\gamma^{-1}(\gamma(\tau)) = \tau$ for all sufficiently complete τ and that $\gamma(\gamma^{-1}(\Pi_1, \ldots, \Pi_n)) = (\Pi_1, \ldots, \Pi_n)$ for all $\Pi_1, \ldots, \Pi_n$ that are pairwise disjoint and have a non-empty union. □

B.2 Fresh Features

Lemma 27. *Let $g \in \mathcal{F}$ and a finite set $F \subseteq \mathcal{F}$ such that $g \notin F$. Then there exists a mapping $\theta : \mathcal{F} \to \mathcal{F}$, which is one-to-one, does not map onto g, such that δ restricted to F is the identity function on F.*

Proof. Since $\mathcal{F}$ is countably infinite there exists an enumeration of $\mathcal{F}$ say $\mathcal{F} = \{f_i \mid i \geq 1\}$ is such an enumeration. Let n be the maximal index of a feature in $F \cup \{g\}$ in this enumeration, *i.e.*, $n = \max\{i \mid f_i \in F \cup \{g\}\}$, which exists since F is finite. We define θ by the following equation:

$$\theta(f) = \begin{cases} f_{n+1} & \text{if } f = g \\ f & \text{if } f = f_i,\ 1 \leq i \leq n, \text{ and } f \neq g. \\ f_{n+i+1} & \text{if } f = f_i \text{ and } i \geq n+1 \end{cases}$$

The function θ is well defined because $g \notin F$ and because $\mathcal{F}$ is infinite. It is obvious that θ is one-to-one, does not map onto g, and leaves F invariant. □

Lemma 28. *Let Φ be a first-order formula over ordering constraints, α a variable assignment into and $\theta : \mathcal{F} \to \mathcal{F}$ a function that is one to one and leave $\mathcal{F}(\Phi)$ invariant. Then α is a solution of Φ if and only if $\theta \circ \alpha$ is a solution of Φ.*

Proof. It is obvious that α is a solution of Φ if and only if $\theta \circ \alpha$ is a solution of $\theta(\Phi)$. Since θ leaves the features in Φ invariant we have $\theta(\Phi) = \Phi$. □

Lemma 29 Fresh Features. *Let $g \notin \mathcal{F}(\varphi \wedge \varphi')$ and $\mathcal{V}(\exists \bar{x} \varphi' \subseteq \mathcal{V}(\varphi)$. If every solution α of φ with $g \notin \mathcal{F}(\alpha(x))$ for all $x \in \mathcal{V}(\varphi)$ is a solution of $\exists \bar{x} \varphi$ then $\varphi \models_{\mathrm{FT}_\leq} \exists \bar{x} \varphi'$ holds.*

Proof. Let $\theta : \mathcal{F} \to \mathcal{F}$ be a mapping that is one-to-one, does not map onto g, and leaves $\mathcal{F}(\varphi)$ invariant (Lemma 27). Suppose that α is a solution of φ over $\mathrm{FT}_\leq$. Hence, $\theta \circ \alpha$ is a solution of φ over $\mathrm{FT}_\leq$ (Lemma 28) that satisfies $g \notin \mathcal{F}(\alpha(x))$ for all $x \in \mathcal{V}(\varphi)$. By assumption, $\theta \circ \alpha$ is a solution of $\exists \bar{x} \varphi'$ over $\mathrm{FT}_\leq^-$. Thus, α is a solution of $\exists \bar{x} \varphi'$ over $\mathrm{FT}_\leq$ (Lemma 28). □

B.3 Entailment in $\mathrm{FT}_\leq^-$ Implies Entailment in $\mathrm{FT}_\leq$

Lemma 16 Assume $g \notin \mathcal{F}(\alpha(x))$ for x and $g \notin \mathcal{F}(\varphi)$. If α is a solution of φ in $\mathrm{FT}_\leq$ then $\delta_g \circ \alpha$ is a solution of φ in $\mathrm{FT}_\leq^-$.

Proof. We have to show that every basic constraint in φ is satisfied by $\delta_g \circ \alpha$.

1. Case $x[f]y$ *in* φ where $f \neq g$ due to $g \notin \mathcal{F}(\varphi)$. We have to verify for all π that $f\pi \in D_{\delta_g(\alpha(x))}$ is equivalent to $\pi \in D_{\delta_g \circ \alpha(y)}$. This is proved by the following sequence of equivalences:

$$f\pi \in D_{\delta_g(\alpha(x))} \quad \text{iff} \quad f\pi \in D_{\alpha(x)} \cup \{\pi' g \mid \pi' \in D_{\alpha(x)}\}$$

Note that $f\pi = \pi' g$ and $f \neq g$ implies the existence of π'' such that $\pi = \pi'' g$ and $f\pi'' = \pi'$. Hence

$$\begin{array}{lll} f\pi \in D_{\delta_g(\alpha(x))} & \text{iff} & \pi \in D_{\alpha(y)} \cup \{\pi'' g \mid f\pi'' \in D_{\alpha(x)}\} \\ & \text{iff} & \pi \in D_{\alpha(y)} \cup \{\pi'' g \mid \pi'' \in D_{\alpha(y)}\} \\ & \text{iff} & \pi \in D_{\delta_g(\alpha(y))} \end{array}$$

The reasoning for the labeling function is similar.

2. Case $x \leq y$ *in* φ. We have to verify the domain inclusion $D_{\delta_g(\alpha(x))} \subseteq D_{\delta_g(\alpha(y))}$.

$$\begin{aligned} D_{\delta_g(\alpha(x))} &= D_{\alpha(x)} \cup \{\pi' g \mid \pi' \in D_{\alpha(x)}\} \\ &\subseteq D_{\alpha(y)} \cup \{\pi' g \mid \pi' \in D_{\alpha(y)}\} \\ &= D_{\delta_g(\alpha(y))} \end{aligned}$$

The reasoning for the labeling function is again similar.

3. The case $a(x)$ *in* φ is simple, since no label is deleted from $L_{\alpha(x)}$. □

Lemma 17 If $g \notin \mathcal{F}(\tau)$ then $\delta_g^{-1}(\delta_g(\tau)) = \tau$.

Proof. Since $g \notin \mathcal{F}(\tau)$, the tree $\delta_g(\tau)$ is well-defined and hence $\delta_g^{-1}(\delta_g(\tau))$ is.

$$\begin{aligned} D_{\delta_g^{-1}(\delta_g(\tau))} &= D_{\delta_g} \setminus \{\pi g \pi' \mid \pi, \pi' \in \mathcal{F}^*\} \\ &= (D_\tau \cup \{\pi g \mid \pi \in D_\tau\}) \setminus \{\pi g \pi' \mid \pi, \pi' \in \mathcal{F}^*\} \\ &= D_\tau \end{aligned}$$

The last equality holds, since we have require $g \notin \mathcal{F}(\tau)$. The argument for the labeling function is symmetric.

$$\begin{aligned} L_{\delta_g^{-1}(\delta_g(\tau))} &= L_{\delta_g(\tau)} \setminus \{(\pi, a) \mid \pi = \pi' g \pi'',\ a \in \mathcal{L}\} \\ &= (L_\tau \cup \{(\pi g, b) \mid \pi \in D_\tau\}) \setminus \{\pi g \pi' \mid \pi, \pi' \in \mathcal{F}^*\} \\ &= L_\tau \end{aligned}$$

□

Lemma 18 Let $g \notin \mathcal{F}(\varphi)$. If α is a solution of φ in $\mathrm{FT}_{\leq}^{-}$ then $\delta_g^{-1} \circ \alpha$ is a solution of φ in $\mathrm{FT}_{\leq}$.

Proof. We have to show that every basic constraint in φ is satisfied by $\delta_g^{-1} \circ \alpha$.

1. Case $x[f]y$ *in* φ where $f \neq g$ due to $g \notin \mathcal{F}(\varphi)$. We have to verify for all π that $f\pi \in D_{\delta_g^{-1}(\alpha(x))}$ is equivalent to $\pi \in D_{\delta_g^{-1}(\alpha(y))}$. This is proved by the following equivalences:

$$\begin{array}{lll} f\pi \in D_{\delta_g^{-1}(\alpha(x))} & \text{iff} & f\pi \in D_{\alpha(x)} \setminus \{f\pi g \pi' \mid \pi, \pi' \in \mathcal{F}^*\} \\ & \text{iff} & \pi \in D_{\alpha(y)} \setminus \{\pi g \pi' \mid \pi, \pi' \in \mathcal{F}^*\} \\ & \text{iff} & \pi \in D_{\delta_g^{-1}(\alpha(y))} \end{array}$$

The reasoning for the labeling function is similar.

2. Case $x{\leq}y$ *in* φ. We have to verify the inclusions $D_{\delta_g^{-1}(\alpha(x))} \subseteq D_{\delta_g^{-1}(\alpha(y))}$ and $L_{\delta_g^{-1}(\alpha(x))} \subseteq L_{\delta_g^{-1}(\alpha(y))}$, which is both obvious.
3. The case $a(x)$ *in* φ is simple, since no label is deleted at the root of some tree $\alpha(x)$. $\square$

B.4 The Final Case Distinction

Lemma 30. *Suppose* $\mathcal{V}(\exists\bar{x}\varphi') \subseteq \mathcal{V}(\varphi)$*,* $\mathcal{V}(\varphi) \cap \mathcal{V}(\bar{x}) = \emptyset$ *and* $g \notin \mathcal{F}(\varphi \wedge \varphi')$*. If* α *is a solution of* $\eta_g(\varphi)$ *over* $\mathrm{FT}^-_{\leq}$ *then* α' *(as defined in the proof of Proposition 24) is a solution of* φ' *over* $\mathrm{FT}^-_{\leq}$ *which coincides with* α *on the variables in* $\mathcal{V}(\varphi)$*.*

1. Case $x{\leq}y$ *in* φ'. We have to show that $(\pi, a) \in L_{\alpha'(x)}$ implies $(\pi, a) \in L_{\alpha'(y)}$.
   (a) Case $x, y \in \mathcal{V}(\bar{x})$.
      L1 If $(\pi, a) \in L_{\alpha'(x)}$ has been added because of $\varphi' \vdash x[\pi]_{\geq}a$ then $\varphi' \vdash y[\pi]_{\geq}a$ such that $(\pi, a) \in L_{\alpha'(y)}$.
      L2 If $(\pi, a) \in L_{\alpha'(x)}$ has been added because of $\pi = \pi' g$, $a = b$, $\varphi' \vdash x[\pi']_{\geq}z$ then also $\varphi' \vdash y[\pi']_{\geq}z$ such that $(\pi' g, b) = (\pi, a) \in L_{\alpha'(y)}$.
      L3 Let $(\pi, a) \in L_{\alpha'(x)}$ because there exists $z \in \mathcal{V}(\varphi)$, z and π_0, π_1, π_2 such that $\pi = \pi_1\pi_2$, $\varphi' \vdash x[\pi_1]_{\geq}z'$, $\varphi' \vdash z[\pi_0]_{\leq}z'$, and $(\pi_0\pi_2, a) \in L_{\alpha(z)}$. Since $x{\leq}y$ *in* φ' we also have $\varphi' \vdash y[\pi_1]_{\geq}z'$ and hence $(\pi_1\pi_2, a) \in L_{\alpha'(y)}$.
   (b) Case $x \in \mathcal{V}(\bar{x})$ and $y \in \mathcal{V}(\varphi)$.
      L1 Let $(\pi, a) \in \alpha'(x)$ because $\varphi' \vdash x[\pi]_{\geq}a$. In this case, $\varphi' \vdash y[\pi]_{\geq}a$. The correctness of this relation implies $\varphi' \models_{\mathrm{FT}_{\leq}} y[\pi]_{\geq}a$. Our assumption of entailment with respect to $\mathrm{FT}_{\leq}$ and $y \in \mathcal{V}(\varphi)$ yield:

      $$\varphi \models_{\mathrm{FT}_{\leq}} \exists\bar{x}\varphi' \models_{\mathrm{FT}_{\leq}} y[\pi]_{\geq}a$$

      Since α is a solution of $\eta_g(\varphi)$ it is also a solution of φ such that $(\pi, a) \in L_{\alpha(y)} = L_{\alpha'(y)}$.
      L2 Let $(\pi g, b) \in \alpha'(x)$ because $\varphi' \vdash x[\pi]_{\geq}z$. Symmetrically to the previous case we can show that $\varphi \models_{\mathrm{FT}_{\leq}} \exists z(y[\pi]_{\geq}z)$. Proposition 22 implies the existence of $z' \in \mathcal{V}(\varphi)$ such that $\varphi \vdash y[\pi]_{\geq}z'$. Since $z'[g]b$ *in* $\eta_g(\varphi)$, we conclude $\eta_g(\varphi) \models y[\pi g]_{\geq}b$. Finally, α is a solution of $\eta_g(\varphi)$ such that $(\pi g, b) \in L_{\alpha(y)} = L_{\alpha'(y)}$.
      L3 Let $(\pi, a) \in L_{\alpha'(x)}$ because there exist $z \in \mathcal{V}(\varphi)$, z', π_0, π_1, π_2 such that $\pi = \pi_1\pi_2$, $\varphi' \vdash x[\pi_1]_{\geq}z'$, $\varphi' \vdash z[\pi_0]_{\leq}z'$, and $(\pi_0\pi_2, a) \in L_{\alpha(z)}$. In this case, $\varphi' \vdash y[\pi_1]_{\geq}z$ and thus $\varphi' \models_{\mathrm{FT}_{\leq}} \exists z' (z[\pi_0]_{\leq}z' \wedge y[\pi_1]_{\geq}z')$. Since $y, z \in \mathcal{V}(\varphi)$, we have:

      $$\varphi \models_{\mathrm{FT}_{\leq}} \exists\bar{x}\varphi' \models_{\mathrm{FT}_{\leq}} \exists z' (z[\pi_0]_{\leq}z' \wedge y[\pi_1]_{\geq}z')$$

      Lemma 23, assumption $(\pi_0\pi_2, a) \in L_{\alpha(z)}$, and that α is a solution of φ imply $(\pi_1\pi_2, a) \in L_{\alpha(y)}$, *i.e.*, $(\pi, a) = L_{\alpha'(y)}$.
   (c) Case $x \in \mathcal{V}(\varphi)$ and $y \in \mathcal{V}(\bar{x})$. If $(\pi, a) \in L_{\alpha'(x)}$ then $(\pi, a) \in L_{\alpha(x)}$. Since $\varphi' \vdash y[\varepsilon]_{\geq}x$, $\varphi' \vdash x[\varepsilon]_{\leq}x$ and $x \in \mathcal{V}(\varphi)$ L3 implies $(\varepsilon\pi, a) \in L_{\alpha'(y)}$.

(d) If $x,y \in \mathcal{V}(\varphi)$ then $\varphi \models_{\mathrm{FT}_\leq} x{\leq}y$ such that $L_{\alpha(x)} \subseteq L_{\alpha(y)}$ and hence $L_{\alpha'(x)} \subseteq L_{\alpha'(y)}$.

2. Case $x[f]y$ *in* φ'. We have to show that $(f\pi, a) \in L_{\alpha'(x)}$ if and only if $(\pi, a) \in L_{\alpha'(y)}$. We first assume $(f\pi, a) \in L_{\alpha'(x)}$ and prove $(\pi, a) \in L_{\alpha'(y)}$.

(a) Case $x,y \in \mathcal{V}(\bar{x})$.

L1 Let $(\pi, a) \in L_{\alpha'(y)}$ because $\varphi' \vdash y[\pi']_{\geq}a$. Hence, $\varphi' \vdash x[f\pi]_{\geq}a$ such that $(f\pi, a) \in \alpha'(x)$.

L2 Let $(\pi, a) \in L_{\alpha'(x)}$ because there exist π and z such that $\pi = \pi' g$, $a = b$ and $\varphi' \vdash y[\pi']_{\geq}z$. Hence, $\varphi' \vdash x[f\pi']_{\geq}z$ such that $(f\pi' g, b) \in L_{\alpha'(x)}$, *i.e.*, $(f\pi, a) \in L_{\alpha'(x)}$.

L3 Let $(\pi, a) \in L_{\alpha'(y)}$ because there exists $z \in \mathcal{V}(\varphi)$, z', π_0, π_1, and π_2 such that $\varphi' \vdash y[\pi_1]_{\geq}z'$, $\varphi' \vdash z[\pi_0]_{\leq}z'$, $(\pi_1\pi_2, a) \in L_{\alpha(z)}$, and $\pi = \pi_1\pi_2$. Since $x[f]y$ we also have $\varphi' \vdash x[f\pi_1]_{\leq}z'$ which yields $(f\pi_1\pi_2, a) \in L_{\alpha'(x)}$, *i.e.*, $(f\pi, a) \in L_{\alpha'(x)}$.

(b) Case $x \in \mathcal{V}(\bar{x})$ and $y \in \mathcal{V}(\varphi)$. If $(\pi, a) \in L_{\alpha'(y)}$ then $(\pi, a) \in L_{\alpha(y)}$. By applying L3 with $z = z' = y$, $\pi_0 = \varepsilon$, $\pi_1 = f$, and $\pi_2 = \pi$, we obtain $(f\pi, a) \in L_{\alpha'(x)}$.

(c) Case $x \in \mathcal{V}(\varphi)$ and $y \in \mathcal{V}(\bar{x})$. Let $(\pi, a) \in L_{\alpha'(y)}$.

L1 If $\varphi' \vdash y[\pi]_{\geq}a$ then $\varphi' \vdash x[f\pi]_{\geq}a$. Since $x \in \mathcal{V}(\varphi)$ this yields $\varphi \models_{\mathrm{FT}_\leq} x[\pi]_{\geq}a$ such that $(f\pi, a) \in L_{\alpha(x)}$, *i.e.*, $(f\pi, a) \in L_{\alpha'(x)}$.

L2 Let $(\pi, a) \in L_{\alpha'(x)}$ because there exists π' and z such that $\varphi' \vdash y[\pi']_{\geq}z$, $\pi = \pi' g$ and $a = b$. Hence $\varphi' \vdash x[f\pi']_{\geq}z$ such that $\varphi \models_{\mathrm{FT}_\leq} \exists z(x[f\pi']_{\geq}z)$ since $x \in \mathcal{V}(\varphi)$. Proposition 22 implies the existence of $z' \in \mathcal{V}(\varphi)$ such that $\varphi \vdash x[f\pi]_{\geq}z'$ and thus $\eta_g(\varphi) \vdash x[f\pi g]_{\geq}b$. Since α is a solution of $\eta_g(\varphi)$, we have $(f\pi' g, b) \in L_{\alpha(x)}$, *i.e.*, $(f\pi, a) \in L_{\alpha'(x)}$.

L3 Let $(\pi, a) \in L_{\alpha'(y)}$ because there exists $z \in \mathcal{V}(\varphi)$, z', π_0, π_1, and π_2 such that $\varphi' \vdash y[\pi_1]_{\geq}z'$, $\varphi' \vdash z[\pi_0]_{\leq}z'$, $(\pi_0\pi_2, a) \in L_{\alpha(z)}$, and $\pi = \pi_1\pi_2$. Since $x[f]y$ we also have $\varphi' \vdash x[f\pi_1]_{\geq}z'$. Since $x,z \in \mathcal{V}(\varphi)$ this implies $\varphi \models_{\mathrm{FT}_\leq} \exists z'(z[\pi_0]_{\leq}z' \wedge x[f\pi_1]_{\geq}z')$. Now, Lemma 23 and $(\pi_0\pi_2, a) \in L_{\alpha(z)}$ imply $(f\pi_1\pi_2, a) \in L_{\alpha(x)}$, *i.e.*, $(f\pi, a) \in L_{\alpha'(x)}$.

(d) Case $x,y \in \mathcal{V}(\varphi)$. In this case $\varphi \models_{\mathrm{FT}_\leq} x[f]y$ such that if $(\pi, a) \in L_{\alpha(y)}$ then $(f\pi, a) \in L_{\alpha(x)}$.

For the converse of the case $x[f]y \in \varphi'$, we assume $(f\pi, a) \in L_{\alpha'(x)}$ and prove $(\pi, a) \in L_{\alpha'(y)}$.

(a) Case $x,y \in \mathcal{V}(\bar{x})$.

L1 Let $(f\pi, a) \in L_{\alpha'(x)}$ because and $\varphi' \vdash x[f\pi']_{\geq}a$. Since φ' is F1F2 closed, this implies $\varphi' \vdash y[\pi]_{\geq}a$ such that $(\pi, a) \in \alpha'(y)$.

L2 Let $(f\pi, a) \in L_{\alpha'(x)}$ because $f\pi = \pi' g$, $a = b$ and $\varphi' \vdash x[\pi']_{\geq}z$. Our assumption $g \notin \mathcal{F}(\varphi')$ implies $f \neq g$ such that there exists π'' with $\pi = \pi'' g$ and $\pi' = f\pi''$. Since φ' is F1F2 closed, $\varphi' \vdash x[f\pi'']_{\geq}z$ and $x[f]y$ *in* φ' imply $\varphi' \vdash y[\pi'']_{\geq}z$. Hence, $(\pi'' g, b) \in L_{\alpha'(y)}$, *i.e.*, $(\pi, a) \in L_{\alpha'(y)}$.

L3 Let $(f\pi, a) \in L_{\alpha'(x)}$ because there exist $z \in \mathcal{V}(\varphi)$, z', and π_0, π_1, π_2 such that $f\pi = \pi_1\pi_2$, $\varphi' \vdash x[\pi_1]_{\geq}z$, $\varphi' \vdash z[\pi_1]_{\leq}z'$, and $(\pi_0\pi_2, a) \in L_{\alpha(z)}$.

A. If $\pi_1 = \varepsilon$ then $f\pi = \pi_2$ and $z'{\leq}x$ *in* φ'. Hence $\varphi' \vdash z[\pi_0 f]_{\geq}y$ and $\varphi' \vdash z[\pi_0 f\pi]_{\geq}a$. Trivially $\varphi' \vdash y[\varepsilon]_{\leq}y$, since $y \in \mathcal{V}(\varphi') \subseteq \mathcal{V}(\varphi)$ such that L3 implies $(\varepsilon\pi, a) \in L_{\alpha(y)}$.

B. Otherwise, $\pi_1 = f\pi_1'$ and $\pi = \pi_1'\pi_2$ for some π'. The F1F2 closedness of φ' implies that $\varphi' \vdash y[\pi_1']_{\leq} z'$. Hence, L3 yields $(\pi_1'\pi_2, a) \in L_{\alpha'(y)}$, *i.e.*, $(\pi, a) \in L_{\alpha'(y)}$.

(b) Case $x \in \mathcal{V}(\bar{x})$ and $y \in \mathcal{V}(\varphi)$.

L1 Let $(f\pi, a) \in L_{\alpha'(x)}$ because $\varphi' \vdash x[f\pi]_{\geq} a$. The F1F2 closedness of φ implies $\varphi' \vdash y[\pi]_{\geq} a$ and hence $\varphi \models_{\mathrm{FT}_{\leq}} y[\pi]_{\geq} a$ since $y \in \mathcal{V}(\varphi)$. For α being a solution of φ this implies $(\pi, a) \in L_{\alpha(y)}$ which is equivalent to $(\pi, a) \in L_{\alpha'(y)}$.

L2 Let $(f\pi, a) \in L_{\alpha'(x)}$ because $f\pi = \pi' g$, $a = b$ and $\varphi' \vdash x[\pi']_{\geq} z$. Since $f \neq g$ there exists there exists π'' such that $\pi = \pi'' g$ and $f\pi'' = \pi'$. By F1F2 closedness we obtain $\varphi' \vdash y[\pi'']_{\geq} z$. Hence $\varphi \models_{\mathrm{FT}_{\leq}} \exists z(y[\pi'']_{\geq} z)$. Since φ is F1F2 closed, Proposition 22 implies the existence of $z' \in \mathcal{V}(\varphi)$ such that $\varphi' \vdash y[\pi'']_{\geq} z'$. The definition of $\eta_g(\varphi)$ implies $\varphi \vdash y[\pi' g]_{\geq} b$. Hence, $(\pi'' g, b) \in L_{\alpha(y)}$ which is equivalent to $(\pi, a) \in L_{\alpha'(y)}$.

L3 Let $(f\pi, a) \in L_{\alpha'(x)}$ because there exist $z \in \mathcal{V}(\varphi)$, z' and π_0, π_1, π_2 such that $f\pi = \pi_1\pi_2$, $\varphi' \vdash x[\pi_1]_{\geq} z'$, $\varphi' \vdash z[\pi_0]_{\leq} z'$, and $(\pi_0\pi_2, a) \in L_{\alpha(z)}$.

A. Case $\pi_1 = \varepsilon$. Hence $f\pi = \pi_2$ and $z' {\leq} x$ *in* φ'. In this case $\varphi' \vdash z[\pi_0 f]_{\leq} y$ such that $z, y \in \mathcal{V}(\varphi)$ implies

$$\varphi \models_{\mathrm{FT}_{\leq}} \exists \bar{x} \varphi' \models_{\mathrm{FT}_{\leq}} z[\pi_0 f]_{\leq} y$$

Since α is a solution of φ and $(\pi_0 f\pi, a) \in L_{\alpha(z)}$. Lemma 23 implies $(\pi, a) \in L_{\alpha(y)}$, *i.e.*, $(\pi, a) \in L_{\alpha'(y)}$.

B. Otherwise there exists π_1' such that $\pi_1 = f\pi_1'$ and $\pi = \pi_1'\pi_2$. The F1F2 closedness of φ' and $\varphi' \vdash x[f\pi_1']_{\geq} z'$ imply $\varphi' \vdash y[\pi_1']_{\geq} z'$. Since $y, z \in \mathcal{V}(\varphi)$ we know that

$$\varphi \models_{\mathrm{FT}_{\leq}} \exists \bar{x} \varphi' \models_{\mathrm{FT}_{\leq}} \exists z' \left(z[\pi_0]_{\geq} z' \wedge y[\pi_1']_{\leq} z'\right)$$

Hence, our assumption $(\pi_0\pi_2, a) \in L_{\alpha(z)}$ and Lemma 23 imply $(\pi_1'\pi_2, a) \in \alpha(y)$, *i.e.*, $(\pi, a) \in \alpha'(y)$.

(c) Case $x \in \mathcal{V}(\varphi)$ and $y \in \mathcal{V}(\bar{x})$. If $(f\pi, a) \in L_{\alpha'(x)}$ then $(f\pi, a) \in L_{\alpha(x)}$. L3, $y \notin \mathcal{V}(\varphi)$, and $x \in \mathcal{V}(\varphi)$ imply $(\pi, a) \in L_{\alpha'(y)}$.

(d) Case $x, y \in \mathcal{V}(\varphi)$. In this case $\varphi \models_{\mathrm{FT}_{\leq}} x[f]y$ such that if $(f\pi, a) \in L_{\alpha(x)}$ then $(\pi, a) \in L_{\alpha(y)}$.

3. Case $a(x)$ *in* φ'.

(a) If $x \in \mathcal{V}(\bar{x})$ then $\varphi' \vdash x[\varepsilon]_{\geq} a$ such that $(\varepsilon, a) \in L_{\alpha'(x)}$ (condition L1).

(b) If $x \in \mathcal{V}(\varphi)$ then $\varphi \models_{\mathrm{FT}_{\leq}} \exists \bar{x} \varphi' \models_{\mathrm{FT}_{\leq}} a(x)$. Hence $(\varepsilon, a) \in \alpha(x)$, *i.e.*, $(\varepsilon, a) \in \alpha'(x)$. □