Temporal Logics over Unranked Trees

Pablo Barceló
University of Toronto
pablo@cs.toronto.edu

Leonid Libkin
University of Toronto
libkin@cs.toronto.edu

Abstract

We consider unranked trees, that have become an active subject of study recently due to XML applications, and characterize commonly used fragments of first-order (FO) and monadic second-order logic (MSO) for them via various temporal logics. We look at both unordered trees and ordered trees (in which children of the same node are ordered by the next-sibling relation), and characterize Boolean and unary FO and MSO queries. For MSO Boolean queries, we use extensions of the μ-calculus: with counting for unordered trees, and with the past for ordered. For Boolean FO queries, we use similar extensions of CTL**. We then use composition techniques to transfer results to unary queries. For the ordered case, we need the same logics as for Boolean queries, but for the unordered case, we need to add both past and counting to the μ-calculus and* CTL**. We also consider MSO sibling-invariant queries, that can use the sibling ordering but do not depend on the particular one used, and capture them by a variant of the μ-calculus with modulo quantifiers.*

1 Introduction

Logics and automata over *unranked* trees – that is, trees in which nodes can have arbitrary many children – have recently received much attention due to XML applications, which typically model XML documents as labeled unranked trees [31]. Logical formalisms for XML languages are usually based on MSO, monadic second-order logic, or FO, first-order logic. While having MSO or FO as their basis, syntactically they could be quite different: e.g., syntactic modifications of MSO or FO that ensure faster query evaluation [30, 32], restrictions of Datalog [15, 16], or model-theoretic formalisms [23]. MSO is often used as a basis for query languages and is closely connected to schema specifications, while navigational aspects (e.g., XPath) are usually based on FO.

It is known that over *ranked* trees (that is, trees in which the number of children is fixed) one has tight connections between logics such as MSO and FO and temporal logics. For example, over the infinite binary tree MSO equals the μ-calculus L_μ [12, 33]. Furthermore, the monadic path logic over the infinite binary tree has precisely the power of CTL* [18], which implies that CTL* = FO over finite binary trees.

Thus, it seems natural to consider analogs of temporal logics over unranked trees and relate them to FO and MSO. Unlike results of [37, 20, 29] that describe fragments of FO and MSO corresponding to modal and temporal logics, we are interested in extensions of temporal logics that have the power of MSO and FO over unranked trees. The reason is that temporal logics define bisimulation-invariant properties, but many XML queries of interest are not bisimulation-invariant (for example, XML DTDs are essentially equivalent to full MSO [39]). Many navigational properties of importance in the XML context are very natural to express in temporal logics; furthermore, temporal logics enjoy nice algorithmic properties which could hopefully be adapted in the XML context. Various connections have been explored in the literature (for example, implication for various CTL fragments based on tree-patterns [28], verifying properties of paths in XML documents [1], ambient logic-based languages for semi-structured data [6], extensions of XPath with an until-like operator [26], or extensions of LTL for trees [35]). Our goal here is to systematically relate FO and MSO over unranked trees to temporal logics.

In the XML setting, one most often studies Boolean queries, giving a yes/no answer on a tree (for example, validation of XML documents with respect to DTDs), or unary queries, selecting nodes from a document (e.g., finding the set of nodes reachable by an XPath expression). Both are naturally modeled by temporal logics which are evaluated in an element in a structure. They naturally define unary queries, and for the Boolean case, we evaluate them at the root.

In view of [12, 20, 33, 18, 29], it is expected that MSO should correspond to some variation of the μ-calculus, and FO should be related to CTL*. For exact statements of results, it is important to decide how precisely we represent trees as transition systems. We shall certainly use the edge relation itself (that is, the child relation $\prec_{\text{ch}}$ in a tree). In unranked trees, one typically has a next-sibling relation $\prec_{\text{sb}}$ on children of the same node. These two will be our basic vocabulary symbols when we deal with MSO. For the case of FO, notice that temporal logics naturally talk about reachability which is not FO-definable. Hence, when dealing with FO, we shall be using the descendant relation $\prec^*_{\text{ch}}$ (which is the transitive closure of $\prec_{\text{ch}}$) and the sibling ordering $\prec^*_{\text{sb}}$ (which is the transitive closure of $\prec_{\text{sb}}$).

Organization The paper is organized as follows. We present notations in Section 2. In Section 3 we deal with Boolean queries. We first review known results relating MSO and FO over unordered trees to L_μ and CTL* with counting, and give a simple direct proof for the MSO case. Then we turn to ordered trees and characterize MSO and FO by adding the ability to reason about the past to L_μ and CTL*. In Section 4 we deal with unary queries. Our proofs are based on the results for Boolean queries and several composition results that allow us to transfer results to the unary case. It turns out that for the ordered case, the same logics work for unary queries, but for unordered trees one has to add past connectives as well. Then in Section 5 we deal with sibling-invariant queries, and show that they are captured by adding modulo quantifiers to L_μ. As a corollary, we get a simple proof of a result by Courcelle. In Section 6 we offer remarks on the complexity of model-checking and on XML applications.

2 Preliminaries

Trees and strings An *unranked tree domain* D is a prefix-closed finite set of strings of natural numbers, such that if $s \cdot i \in D$ then $s \cdot j \in D$, for all $j < i$. If Σ is a finite alphabet, then an (ordered) *unranked* Σ*-tree* is a first-order structure

$$T \;=\; \langle D, \prec_{\text{ch}}, \prec_{\text{sb}},\ \prec^*_{\text{ch}}, \prec^*_{\text{sb}},\ (P_a)_{a \in \Sigma}\rangle,$$

where $\prec_{\text{ch}}$ is interpreted as the child relation ($s \prec_{\text{ch}} s \cdot i$ if $s, s \cdot i \in D$), $\prec_{\text{sb}}$ as the sibling order ($s \cdot i \prec_{\text{sb}} s \cdot (i+1)$ if $s \cdot (i+1), s \cdot i \in D$), and P_a as the set of nodes labeled a, for $a \in \Sigma$ (as usual, we require that the P_a's form a partition of D, that is, each element of the domain is assigned a unique label in Σ). Furthermore, $\prec^*_{\text{ch}}$ is the transitive-reflexive closure of $\prec_{\text{ch}}$ (the descendant relation), and $\prec^*_{\text{sb}}$ is the transitive-reflexive closure of $\prec_{\text{sb}}$ (linear ordering on siblings). The empty string will be denoted by ε; hence $\varepsilon \in D$ is the root of T.

We shall denote first-order logic by FO and monadic second-order logic (that extends FO with quantification over sets) by MSO. In all our applications, the vocabulary will contain at least the labeling predicates P_a for $a \in \Sigma$. The rest of the vocabulary will be explicitly listed in square brackets; for example, MSO[$\prec_{\text{ch}}, \prec_{\text{sb}}$] refers to MSO over the vocabulary $(\prec_{\text{ch}}, \prec_{\text{sb}}, (P_a))$, and FO[$\prec^*_{\text{ch}}$] refers to FO over the vocabulary $(\prec^*_{\text{ch}}, (P_a))$. Since $\prec^*_{\text{ch}}$ and $\prec^*_{\text{sb}}$ are definable from $\prec_{\text{ch}}$ and $\prec_{\text{sb}}$ in MSO but not in FO, we normally use $\prec_{\text{ch}}$ and $\prec_{\text{sb}}$ for MSO, and $\prec^*_{\text{ch}}$ and $\prec^*_{\text{sb}}$ for FO.

We shall view a finite string s of length n over Σ as a structure $\langle [n], <, (P_a)_{a \in \Sigma}\rangle$ with the universe $[n] = \{0, \ldots, n-1\}$, a binary predicate $<$ interpreted as the usual ordering, and P_a interpreted as the set of positions $i < n$ in which $a \in \Sigma$ occurs.

Unranked tree automata An *unranked nondeterministic tree automaton (UNTA)* is $\mathcal{A} = (Q, \Sigma, F, \delta)$, where Q is a set of states, Σ is the alphabet, $F \subseteq Q$ is the set of final states, and $\delta : Q \times \Sigma \to 2^{Q^*}$ is the transition function such that for every $q \in Q$ and $a \in \Sigma$, $\delta(q, a)$ is a regular language over Q.

A *run* of an automaton $\mathcal{A}$ on a tree T with domain D is a function $\rho : D \to Q$ such that, for every element $s \in D$ labeled a with children $s_1 \prec_{\text{sb}} \ldots \prec_{\text{sb}} s_n$, the string $\rho(s_1)\rho(s_2)\ldots\rho(s_n)$ is in $\delta(\rho(s), a)$. A tree is *accepted* by $\mathcal{A}$ if there is a run ρ such that $\rho(\varepsilon) \in F$ (that is, the root is in a state in F). The set of trees accepted by $\mathcal{A}$ is denoted by $L(\mathcal{A})$. An automaton $\mathcal{A}$ is *deterministic* if for every tree it has exactly one run.

Temporal logics We shall mostly consider the modal μ-calculus L_μ and CTL*. These will be interpreted over finite *transition systems* $\mathcal{K} = \langle S, (E_r)_{r \in R}, (P_a)_{a \in \Sigma}\rangle$, where R is a finite list of binary relation symbols and each E_r is interpreted as a subset of $S \times S$; each P_a is a subset of S. We shall view a Σ-tree T as a transition system with two binary relations $\prec_{\text{ch}}$ and $\prec_{\text{sb}}$.

The formulae of L_μ are given by

$$\varphi := a\ (a \in \Sigma) \mid X \mid \varphi \vee \varphi \mid \neg\varphi \mid \Diamond(E_r)\varphi \mid \mu X\ \varphi(X),$$

where in $\mu X\ \varphi(X)$, the variable X must occur positively in φ. Given $\mathcal{K}$, $s \in S$, and a valuation v for free variables (such that each $v(X)$ is a subset of S), we define the semantics (omitting the rules for propositional letters and Boolean connectives) by

- $(\mathcal{K}, v, s) \models X$ iff $s \in v(X)$.
- $(\mathcal{K}, v, s) \models \Diamond(E_r)\varphi$ iff $(\mathcal{K}, v, s') \models \varphi$ for some s' with $(s, s') \in E_r$.
- $(\mathcal{K}, v, s) \models \mu X\, \varphi(X)$ iff s is in the least fixed point of the operator defined by φ; in other words, if $s \in \bigcap\{P \mid \{s' \mid (\mathcal{K}, v[P/X], s') \models \varphi\} \subseteq P\}$, where $v[P/X]$ extends the valuation v by $v(X) = P$.

When we refer to L_μ being equivalent to a logic on trees, we mean L_μ formulae without free variables (which are then evaluated in an element of $\mathcal{K}$). As usual, we define $\Box(E_r)\varphi$ as $\neg\Diamond(E_r)\neg\varphi$. If we list explicitly binary relations E_i's, we write $L_\mu[E_1, \ldots, E_k]$ to refer L_μ formulae that only use those relations. For example, $L_\mu[\prec_{\rm ch}, \prec_{\rm sb}]$ refers to L_μ formulae that use both $\prec_{\rm ch}$ and $\prec_{\rm sb}$ relations.

Next we define CTL* in a way that is convenient when we have several binary relations, say $E_1, \ldots, E_m$. Then CTL$^\star[E_1, \ldots, E_m]$ has *state* formulae α, and E_i*-path* formulae β_i, $i \leq m$, defined by the grammars below:

$$\begin{array}{lcl} \alpha & := & a\ (a \in \Sigma)\ \mid\ \neg\alpha\ \mid\ \alpha \vee \alpha\ \mid\ \mathbf{E}\beta_i,\ i \leq m \\ \beta_i & := & \alpha\ \mid \neg\beta_i\ \mid\ \beta_i \vee \beta_i\ \mid\ \mathbf{X}_{E_i}\beta_i\ \mid\ \beta_i\, \mathbf{U}_{E_i}\, \beta_i \end{array}$$

Of course for the case of just one binary relation this is the standard definition of CTL*.

An E_i-path π is a sequence $s_1 s_2 \ldots$ of nodes such that $(s_j, s_{j+1}) \in E_i$ for every s_j, s_{j+1} in π, and such that if the set $\{s \mid (s_j, s) \in E_i\}$ is nonempty, then one of the elements of this set is s_{j+1}. (Note that in trees, both $\prec_{\rm ch}$-paths and $\prec_{\rm sb}$-paths will be finite, although typically in transition systems one considers infinite paths. This is not a problem, however, since we can make all paths infinite by adding a child labeled $\perp \notin \Sigma$ to each leaf, and a $\prec_{\rm ch}$ loop for that element, and likewise for the youngest sibling on each sibling path.)

Given an E_i-path $\pi = s_1 s_2 \ldots$, we let π^k be the path starting at s_k. Then (we only list the essential rules):

- $(\mathcal{K}, s) \models \mathbf{E}\beta_i$ iff there exists an E_i-path $\pi = s \ldots$ such that $(\mathcal{K}, \pi) \models \beta_i$;
- $(\mathcal{K}, \pi) \models \alpha$ iff $(\mathcal{K}, s_1) \models \alpha$, where $\pi = s_1 \ldots$;
- $(\mathcal{K}, \pi) \models \mathbf{X}_{E_i}\beta_i$ iff $\pi = s_1 s_2 \ldots$ is an E_i-path and and $(\mathcal{K}, \pi^2) \models \beta_i$; and
- $(\mathcal{K}, \pi) \models \beta_i\, \mathbf{U}_{E_i}\, \beta_i'$ iff π is an E_i-path and there is a number k such that $(\mathcal{K}, \pi^k) \models \beta_i'$ and $(\mathcal{K}, \pi^l) \models \beta_i$ for all $l < k$.

Sometimes (e.g. [18]), one has a version of CTL* that has different $\mathbf{X}_i$ operators for different binary relations E_i and only one until operator $\mathbf{U}$ for all the relations, which refers to paths over the union of all E_i's. Such a logic, over binary trees, is equivalent to monadic path logic of two successor relations [18]. Hence, over finite binary trees, where each path is uniquely determined by its endpoints, this logic is equivalent to FO. Notice, however, in case of relations $\prec_{\rm ch}$ and $\prec_{\rm sb}$, a path over $\prec_{\rm ch} \cup \prec_{\rm sb}$ is *not* uniquely determined by its endpoints.

We shall refer to this version of CTL* as CTL$^\star[\bigcup_i E_i]$. In this case path formulae are closed under the until operator $\mathbf{U}$ and next operators $\mathbf{X}_{E_i}$, and the semantics of $\mathbf{X}_{E_i}\varphi$ is that the next element on the path is reached by an E_i-edge, and φ is true in the continuation of the path from that element.

Boolean and unary queries Since temporal logic formulae are satisfied in an element of a transition system (or a node of a tree in our case), they naturally give rise to classes of Boolean and unary queries definable over unranked trees. We say that MSO (or FO) unary queries over unranked trees are captured by a temporal logic $\mathcal{L}$ if for every MSO (FO, resp.) formula $\varphi(x)$ there exists an $\mathcal{L}$-formula ψ such that for every unranked tree T and a node s we have $T \models \varphi(s) \Leftrightarrow (T, s) \models \psi$, and conversely, for every $\mathcal{L}$ formula ψ we can find $\varphi(x)$ such that the above holds.

MSO (or FO) Boolean queries over unranked trees are captured by a temporal logic $\mathcal{L}$ if for every MSO (FO, resp.) sentence φ there exists an $\mathcal{L}$-formula ψ such that for every unranked tree T we have $T \models \varphi \Leftrightarrow (T, \varepsilon) \models \psi$, and conversely, for every $\mathcal{L}$ formula ψ we can find a sentence φ such that the above holds. That is, Boolean queries are witnessed at the root.

In general, if one of the above is the case, we shall say that over unranked trees MSO $= \mathcal{L}$ or FO $= \mathcal{L}$, and it will be clear from the context whether we speak of Boolean or unary queries.

Quantifier rank, types The quantifier rank $\mathsf{qr}(\varphi)$ of a formula (FO or MSO) is the depth of quantifier nesting in φ. With each structure $\mathfrak{M}$ of vocabulary θ we associate its FO rank-k type $\mathrm{tp}^k_{\mathrm{FO}[\theta]}(\mathfrak{M}) = \{\varphi \mid \mathfrak{M} \models \varphi \text{ and } \mathsf{qr}(\varphi) = k\}$, where φ ranges over FO sentences. Similarly we define $\mathrm{tp}^k_{\mathrm{MSO}[\theta]}(\mathfrak{M})$. If the vocabulary θ is understood, we omit it (and for trees, we normally omit the unary predicates).

A rank-k (FO) type is a set of sentences of the form $\mathrm{tp}^k_{\mathrm{FO}}(\mathfrak{M})$ (and likewise for MSO). It is well-known that there are finitely many rank-k types for all k, and for each rank-k (FO or MSO) type τ there is a sentence (FO or MSO) φ_τ such that $\mathfrak{M} \models \varphi_\tau$ iff $\mathrm{tp}^k_{\mathrm{FO}}(\mathfrak{M}) = \tau$

(or $\mathrm{tp}^k_{\mathrm{MSO}}(\mathfrak{M}) = \tau$). We normally associate types with formulae that define them.

We write $\mathfrak{M} \equiv_k \mathfrak{M}'$ iff $\mathrm{tp}^k_{\mathrm{FO}}(\mathfrak{M}) = \mathrm{tp}^k_{\mathrm{FO}}(\mathfrak{M}')$ (equivalently, if the duplicator wins the k-round Ehrenfeucht-Fraïssé game on $\mathfrak{M}$ and $\mathfrak{M}'$) and likewise for MSO.

Remarks on proof techniques Standard proof techniques for relating logics like FO and MSO to temporal logics include automata, and the composition method [24, 18, 32]. In addition, some results can be shown by using the standard translation from unranked trees into binary trees. However, these reductions, while often useful steps in proofs, do not usually immediately yield the desired results. For example, the standard translation of unranked Σ-labeled trees into ranked ones introduces several nodes labeled by a symbol $\perp \notin \Sigma$. Hence, over translations, the CTL* formula $\mathbf{EF}\perp$ will be true, but since no node in the original tree is labeled $\perp$ it is unclear how to translate this formula back into CTL* over unranked trees. Another difficulty arises when one analyzes paths in unranked trees that are translations of paths in ranked trees: those can change direction from "child" to "next sibling" arbitrarily many times, which causes problems with translating temporal logic formulae.

3 Boolean queries

Throughout this section, all statements of the form FO $= \mathcal{L}$ or MSO $= \mathcal{L}$ refer to Boolean queries.

3.1 Unordered trees

In unordered unranked trees, one does not have the sibling ordering, so we are dealing with logics MSO$[\prec_{\mathrm{ch}}]$ and FO$[\prec^*_{\mathrm{ch}}]$. We review results below for the sake of completeness and comparison with other results, as they follow from known facts [40, 29]. Furthermore, we shall provide some extensions in Section 4.

Define the *counting μ-calculus* C_μ (cf. [11, 19]) as an extension of L_μ with formulae $\Diamond^{\geq k}(E_r)\varphi$. The semantics of $(\mathcal{K}, s) \models \Diamond^{\geq k}(E_r)\varphi$ is as follows: there exist distinct elements $s_1, \ldots, s_k$ such that $(s, s_i) \in E_r$ and $(\mathcal{K}, s_i) \models \varphi$ for every $1 \leq i \leq k$. Then we have the following consequence of results in: [40, 11, 19]:

Theorem 3.1 *Over unranked trees,* MSO$[\prec_{\mathrm{ch}}]$ $=$ $C_\mu[\prec_{\mathrm{ch}}]$. *Furthermore, the translations between* MSO *and C_μ are effective.*

Proof. The equality can easily be deduced from known results: given an MSO sentence φ, consider $\varphi \wedge \varphi_{\mathbf{T}}$ where $\varphi_{\mathbf{T}}$ says that the structure is a tree. This sentence is clearly invariant under unwinding and thus by [40, 11, 19] is equivalent to a C_μ formula which then is equivalent to φ over unranked trees.

To see effectiveness, we provide (in the appendix) a simple direct proof that translates tree automata capturing MSO$[\prec_{\mathrm{ch}}]$ into C_μ. □

Next, define a similar counting extension of CTL* denoted by CTL$^\star_{\mathrm{count}}$. If α is a state formula, we allow a new state formulae $\mathbf{EX}^{\geq k}_E\alpha$ such that $(\mathcal{K}, s) \models \mathbf{EX}^{\geq k}_E\alpha$ if there exist distinct elements $s_1, \ldots, s_k$ such that $(s, s_j) \in E$ and $(\mathcal{K}, s_j) \models \alpha$ for every $1 \leq j \leq k$.

Theorem 3.2 (Moller, Rabinovich [29]) *Over unranked trees,* FO$[\prec^*_{\mathrm{ch}}]$ $=$ CTL$^\star_{\mathrm{count}}[\prec_{\mathrm{ch}}]$. □

3.2 Ordered trees

In this section we look at the full vocabulary containing both $\prec_{\mathrm{ch}}$ and $\prec_{\mathrm{sb}}$ for the case of MSO, and their transitive closures $\prec^*_{\mathrm{ch}}$ and $\prec^*_{\mathrm{sb}}$ for the case of FO. We show that these are captured by L_μ and CTL* if we add the ability to reason about the past.

3.2.1 MSO

The *full* μ-calculus L^{full}_μ (cf. [38]) adds backward modalities $\Diamond(E^-)\varphi$ with the semantics $(\mathcal{K}, s) \models \Diamond(E^-)\varphi$ iff $(\mathcal{K}, s') \models \varphi$ for some s' such that $(s', s) \in E$.

We shall also consider an additional binary relation *first-child* over unranked trees (as is common in many XML applications and in some languages, see, e.g. [15, 16]). We denote it by $\prec_{\mathrm{fc}}$; then in a tree with domain D, $s \prec_{\mathrm{fc}} s'$ iff s, s' both belong to D and $s' = s \cdot 0$.

Theorem 3.3 *Over unranked trees,*

$$\mathrm{MSO}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] = L^{\mathrm{full}}_\mu[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] = L_\mu[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}, \prec_{\mathrm{fc}}],$$

and translations between these formalisms are effective.

Proof sketch. Translations from $L^{\mathrm{full}}_\mu[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ and $L_\mu[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}, \prec_{\mathrm{fc}}]$ into MSO are routine. To see that MSO$[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ is subsumed by $L_\mu[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}, \prec_{\mathrm{fc}}]$, we consider a deterministic unranked tree automaton $\mathcal{A}_\varphi$ equivalent to an MSO sentence φ, and define a simultaneous fixed point formula that computes the sets

$X_i, i \leq m$ (where m is the number of states of $\mathcal{A}_\varphi$) such that for every tree T with domain D, $s \in X_i$ iff the unique run of $\mathcal{A}_\varphi$ assigns state q_i to s. In the process, we need an inner fixed point computation that checks if the string of states assigned to children $s \cdot 0, \ldots, s \cdot j$ of s is in the right regular language. We can check that such a string is in a given regular language using just the $\Diamond(\prec_{\mathrm{sb}})$ modalities if a formula is evaluated in the first position of a string. Hence, with $\prec_{\mathrm{fc}}$, this can be done in L_μ. Alternatively, with backward modalities we can reach the beginning of the string, and from there check membership in a regular language. Notice also that the $\mathrm{MSO}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] = L_\mu[\prec_{\mathrm{fc}}, \prec_{\mathrm{sb}}]$ equality can be proved by using translation into ranked trees and equality of MSO and L_μ over binary trees [33, 12].

3.2.2 FO

FO over trees was studied in several recent papers, partly due to its close connection with XPath [2, 5, 3, 25, 26]. Some of these papers provide algebraic characterizations of FO and its fragments on trees [5, 3], others define extensions of the XPath formalism to capture FO [25, 26].

To capture FO over $\prec^*_{\mathrm{ch}}$ and $\prec^*_{\mathrm{sb}}$, we shall use an extension $\mathrm{CTL}^\star_{\mathrm{past}}$ of $\mathrm{CTL}^\star$ that allows reasoning about the past. Normally such a logic is defined by allowing in addition to $\mathbf{X}$ and $\mathbf{U}$ their "inverses" usually called $\mathbf{Y}$ (yesterday) and $\mathbf{S}$ (since) [22]. We shall use the notation $\mathbf{X}_{\prec^-_{\mathrm{ch}}}$ and $\mathbf{X}_{\prec^-_{\mathrm{sb}}}$ instead of $\mathbf{Y}$, referring to them as next with respect to inverses of $\prec^-_{\mathrm{ch}}$ and $\prec^-_{\mathrm{sb}}$ of $\prec_{\mathrm{ch}}$ and $\prec_{\mathrm{sb}}$. In general, the semantics of path formulae of $\mathrm{CTL}^\star_{\mathrm{past}}$ refers to a path and a position in a path; that is, one defines the notion $(\mathcal{K}, \pi, \ell) \models \beta$. A path therefore includes not only the future but also the past, and we require that paths include the entire past. That is, all $\prec_{\mathrm{ch}}$-paths start in the root, and all $\prec_{\mathrm{sb}}$-paths start in the oldest child.

The semantics of $\mathrm{CTL}^\star_{\mathrm{past}}[E_1, \ldots, E_r]$ is as follows [22] (again, listing only the essential rules):

- $(\mathcal{K}, s) \models \mathbf{E}\beta_i$ if there is an E_i-path $\pi = s_1 s_2 \ldots$ and $\ell \geq 1$ such that $s = s_\ell$ and $(\mathcal{K}, \pi, \ell) \models \beta_i$;
- $(\mathcal{K}, \pi, \ell) \models \mathbf{X}_{E^-_i} \beta_i$ iff π is an E_i-path, $\ell > 1$ and $(\mathcal{K}, \pi, \ell - 1) \models \beta_i$;
- $(\mathcal{K}, \pi, \ell) \models \beta_i \;\mathbf{S}_{E_i}\; \beta'_i$ iff π is an E_i-path and there exists $k < \ell$ such that $(\mathcal{K}, \pi, k) \models \beta'_i$ and $(\mathcal{K}, \pi, j) \models \beta_i$ for all $k < j \leq \ell$.

We shall use the abbreviation $\mathbf{P}_{E_i}\beta$ for *true* $\mathbf{S}_{E_i}\; \beta$ (sometime in the past β).

One can also define a version of $\mathrm{CTL}^\star_{\mathrm{past}}$ with unique "until" and "since" operators and several "next" and "previous" operators. We shall denote this logic by $\mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}]$. The semantics naturally combines the semantics of past and the the semantics of $\mathrm{CTL}^\star[\bigcup_i E_i]$ (that is, we have $\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}$ paths). For example, $(T, \pi, \ell) \models \mathbf{X}_{\prec_{\mathrm{ch}}}\varphi$ if $(T, \pi, \ell + 1) \models \varphi$ and from position ℓ to position $\ell + 1$ one goes by the child relation.

It is known that over a Kripke structure (or a transition system with a single binary relation), $\mathrm{CTL}^\star_{\mathrm{past}} = \mathrm{CTL}^\star$ if each state has a unique path that leads from an initial state to it [22]. But unranked trees are modeled as transition systems with two binary relations, and over the union of these relations, there may be more than one history. In fact one can easily show that over unranked trees, $\mathrm{CTL}^\star \subsetneq \mathrm{CTL}^\star_{\mathrm{past}}$ (for example, a formula saying that the root's oldest child is labeled b and one other child is labeled a is not expressible in $\mathrm{CTL}^\star$ as can be shown by a simple game argument).

Our main result connecting FO and past $\mathrm{CTL}^\star$ can be seen as an analog of the equality between FO and $\mathrm{CTL}^\star$ over finite binary trees [18]. Furthermore, the result that uses the logic with multiple until operators is a refinement of a result in [26] (we provide additional information on the use of the past connectives).

Theorem 3.4 *Over unranked trees,*

$$\begin{array}{rl} & \mathrm{FO}[\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}] \\ = & \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] \\ = & \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}]. \end{array}$$

Moreover, every formula of $\mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ *is equivalent to a formula that does not use since operators* $\mathbf{S}_{\prec_{\mathrm{ch}}}$ *and* $\mathbf{S}_{\prec_{\mathrm{sb}}}$ *as well as* $\mathbf{X}_{\prec^-_{\mathrm{ch}}}$ *but uses* $\mathbf{X}_{\prec^-_{\mathrm{sb}}}$ *and* $\mathbf{P}_{\prec_{\mathrm{sb}}}$*, and the translation between these logics are are effective.*

Proof sketch. We use the standard translation of unranked trees into binary trees $\tau : D \to \{0, 1\}$ such that $\tau(\varepsilon) = \varepsilon$ and for each string $s = s' \cdot i$ in D, $\tau(s{\cdot}0) = \tau(s){\cdot}0$ and $\tau(s'{\cdot}(i{+}1)) = \tau(s){\cdot}1$. Moreover, we complete $\tau(D)$ to make it a binary tree without unary branching. That is, if $\tau(s)$ has only one child in the image of τ on D, we add the second child and give it a new label $\perp \notin \Sigma$. This extends τ to translation from unranked trees to binary ones. It is well known that if we consider the resulting binary trees as trees in the vocabulary $\prec_0, \prec_1, \prec, (P_a)_{a \in \Sigma}, P_\perp$, where $\prec_0$ and $\prec_1$

are interpreted as the first and second successor and $\prec$ is the prefix relation, then there is an effective translation $\varphi \mapsto \varphi^\circ$ from FO sentences over $(\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}, P_a)$ into FO sentences over $(\prec_0, \prec_1, \prec, (P_a)_{a\in\Sigma}, P_\perp)$ such that $T \models \varphi$ iff $\tau(T) \models \varphi^\circ$.

Suppose we are given a sentence φ of $\mathrm{FO}[\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}]$. Consider φ° over binary $\Sigma \cup \{\perp\}$-labeled trees. By [18], there exists a $\mathrm{CTL}^\star$ formula ψ over $\prec_0, \prec_1$ and atomic propositions $a \in \Sigma$ and $\perp$ such that $\tau(T) \models \varphi^\circ$ iff $\tau(T) \models \psi$.

The main part of the proof is providing a translation of ψ into $\mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$. There are two main complications: first, one has to be careful about about the extra nodes labeled $\perp$ (for example, one cannot translate $\perp$ by *false* because formulae such as $\mathbf{EF}\perp$ will be true in trees $\tau(T)$). Also paths over $\tau(T)$ may change direction from $\prec_0$ to $\prec_1$ arbitrarily many times, and thus correspond to an arbitrary union of $\prec_{\mathrm{ch}}$ and $\prec_{\mathrm{sb}}$ paths in T. But all paths over $\tau(T)$ have the following shape when translated back into T: a path may continue along the $\prec_{\mathrm{sb}}$-relation, then continue along the $\prec_{\mathrm{ch}}$-relation, but then with each element it also includes all its older siblings. Thus, a path formula is translated into a family of pairs consisting of a $\prec_{\mathrm{sb}}$-path formula, and a $\prec_{\mathrm{ch}}$-path formula, with the latter also referring to the past, with respect to the sibling relation, of each element. Very roughly, if β and γ are translated into β' and γ' then $\beta\,\mathbf{U}\,\gamma$ will be translated into a formula of the following shape:

$$\beta' \,\mathbf{U}_{\prec_{\mathrm{sb}}} \left(\beta' \wedge \mathbf{EX}_{\prec_{\mathrm{ch}}}\big((\beta' \wedge \neg\mathbf{P}_{\prec_{\mathrm{sb}}}\neg\beta')\,\mathbf{U}_{\prec_{\mathrm{ch}}}\,\gamma'\big)\right)$$

saying that β' holds on a sibling path, and then on a child path, so that it in addition none of the older siblings witnesses $\neg\beta'$, until γ' holds. This leads to further complication as one needs to reason about finite non-maximal paths, but one can show that such paths can be added without increasing the expressiveness of state formulae. Full details of the translation are shown in the appendix. It also shows that in the resulting formulae we only need to reason about the past $\mathbf{P}$ with respect to the next-sibling relation.

$\mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ can easily be translated into $\mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}]$. In turn, $\mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}]$ can be translated into FO. One way of showing this is by translating it into $\mathrm{CTL}^\star_{\mathrm{past}}$ over trees $\tau(T)$ and then using [18] to conclude that formulae can be expressed in FO. Another way is by mimicking the proof of translation from $\mathrm{CTL}^\star$ over binary trees to deal with multiple changes of path direction from $\prec_{\mathrm{ch}}$ to $\prec_{\mathrm{sb}}$. This shows that all the formalisms are equivalent. $\square$

Notice that we also have an analog of Theorem 3.3 stating that $\mathrm{MSO}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] = L_\mu[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}, \prec_{\mathrm{fc}}]$, as we can see that $\mathrm{FO}[\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}] = \mathrm{CTL}^\star[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}, \prec_{\mathrm{fc}}]$. Indeed, that FO can be embedded into $\mathrm{CTL}^\star[\prec_{\mathrm{sb}}, \prec_{\mathrm{fc}}]$ is an immediate consequence of [18], and since $\mathbf{X}_{\prec_{\mathrm{fc}}}$ can be expressed in $\mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}]$ we get the converse as well.

4 Unary queries

In this section we look at temporal logics capturing MSO and FO unary queries; that is, we need equivalence in every node of the tree, not just the root. We start with the case of ordered trees, where we can use logics already defined, and then move to the unordered case where an extension of logics seen in Section 3 provides the desired characterization.

4.1 Ordered trees

Being able to talk about an arbitrary point in a tree means knowing the entire "history", that is, being able to reason about the past with respect to $\prec_{\mathrm{ch}}$ and $\prec_{\mathrm{sb}}$ relations. This ability is already present in extensions of L_μ and $\mathrm{CTL}^\star$ seen in Section 3, and in fact we can prove the following.

Theorem 4.1 *The equivalences*

$$\begin{array}{rrcl} a) & \mathrm{MSO}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] & = & L^{\mathrm{full}}_\mu[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] \\ b) & \mathrm{FO}[\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}] & = & \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}] \\ & & = & \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}], \end{array}$$

hold for unary queries over unranked trees.

In part *b)*, the first equality is an immediate consequence of results of [26]. There are several ways in which these results can be proved. For example, part *a)* can be proved in a way similar to the proof of Theorem 3.3 by using query automata [32] instead of the usual automata. The proof of [26], which uses an LTL-like logic, is very syntactic and establishes the separation property for the logic.

Here we give more semantic proofs based on the composition method [24, 18, 32]. We provide a sketch for the FO case. Consider an unranked tree T with domain D and $s \in D$. We use the notation T_s for the subtree of T rooted at s. Let $s = s' \cdot (\ell - 1)$ (that is, s is an ℓth child), and let $s' \cdot (p-1)$ be the highest numbered child of s' (that is, s' has p children). Fix a number

k and let $\sigma_1, \ldots, \sigma_t$ enumerate all the quantifier-rank $k+2$ types of $\mathrm{FO}[\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}]$.

We define a string $\bar{w}^{\rightarrow}_k(s)$ of length $p - l + 1$ such that the ith letter of this string is $\mathrm{tp}^{k+2}_{\mathrm{FO}[\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}]}(T_{s' \cdot (\ell - 1 + i)})$. We also define a string $\bar{w}^{\leftarrow}_k(s)$ of length $\ell - 1$ whose ith letter is $\mathrm{tp}^{k+2}_{\mathrm{FO}[\prec^*_{\mathrm{ch}}, \prec^*_{\mathrm{sb}}]}(T_{s' \cdot i})$.

Let $\rho_1, \ldots, \rho_m$ enumerate all the quantifier-rank $k+2$ types of strings over the alphabet $\{\sigma_1, \ldots, \sigma_t\}$, and let $\Gamma = \{\rho_1, \ldots, \rho_m\}$. We define an extended alphabet

$$\Sigma' = \Sigma \times (\Gamma \cup \{\#\}) \times (\Gamma \cup \{\#\}).$$

Now with each element s of the domain D of a tree T at distance d from the root, we associate a string $\bar{w}_k(T, s)$ of length $d+1$ over Σ' as follows. Let $s_0, s_1, \ldots, s_d$ be the path leading from the root s_0 of T to $s = s_d$. Then in $\bar{w}_k(T, s)$ the ith letter is $(a, \rho_{i1}, \rho_{i2})$ where: a is the label of s_i; and if $i < d$, then $\rho_{i1} = \mathrm{tp}^{k+2}_{\mathrm{FO}}(\bar{w}^{\rightarrow}_k(s_{i+1}))$, and $\rho_{i2} = \mathrm{tp}^{k+2}_{\mathrm{FO}}(\bar{w}^{\leftarrow}_k(s_{i+1}))$; if $i = d$, then $\rho_{i1} = \rho_{i2} = \#$. The main composition lemma, proved by an Ehrenfeucht-Fraïssé game argument, is the following.

Lemma 4.2 *Given trees T and T', and elements s and s' in T and T', respectively, the following holds:*

$$\bar{w}_k(T, s) \ \equiv_{k+2} \ \bar{w}_k(T', s') \ \implies \ (T, s) \ \equiv_k \ (T', s')\,.$$

Since the relation $\equiv_k$ is of finite index, every FO formula φ with $\mathsf{qr}(\varphi) = k$ is equivalent to a disjunction of formulae specifying the rank-$(k+2)$ type of $\bar{w}_k(T, s)$. Thus, it remains to show how to define those in $\mathrm{CTL}^{\star}_{\mathrm{past}}$. Since every FO formula over strings is equivalent to an LTL formula by Kamp's theorem [21], one simply codes the sentence defining the type of $\bar{w}_k(T, s)$ into (past) LTL with new alphabet letters defining types of strings $\bar{w}^{\rightarrow}_k(s_i)$ and $\bar{w}^{\leftarrow}_k(s_i)$. Those (again by Kamp's theorem) can be expressed in LTL or past LTL over an alphabet which includes symbols for rank-k types of subtrees rooted in nodes. But by Theorem 3.4 (using the fact that one can eliminate $\prec_{\mathrm{ch}}$-past from $\mathrm{CTL}^{\star}_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$) we can see that each such type is definable in $\mathrm{CTL}^{\star}_{\mathrm{past}}$, finishing the proof.

4.2 Unordered trees

In logics C_μ and $\mathrm{CTL}^{\star}_{\mathrm{count}}$ used for the unordered case we do not have the ability to refer to the past, which is essential if we want to deal with unary queries. Thus, we extend them in the following ways: C^{full}_μ refers to the extension of the full μ-calculus L^{full}_μ with counting, that is, formulae $\Diamond^{\geq k}(E_r)\varphi$. Notice that counting is only done with respect to the future, not the past. Likewise, we define $\mathrm{CTL}^{\star}_{\mathrm{count,\ past}}$ as an extension of $\mathrm{CTL}^{\star}_{\mathrm{past}}$ with formulae $\mathbf{EX}^{\geq k}_{E_i}\varphi$ that counting the number of E_i-successors of a given element satisfying φ (that is, counting again is done with respect to the future). Then we have the following result.

Theorem 4.3 *The equivalences*

$$\begin{array}{rrcl} a) & \mathrm{MSO}[\prec_{\mathrm{ch}}] & = & C^{\mathrm{full}}_\mu[\prec_{\mathrm{ch}}] \\ b) & \mathrm{FO}[\prec^*_{\mathrm{ch}}] & = & \mathrm{CTL}^{\star}_{\mathrm{count,\ past}}[\prec_{\mathrm{ch}}] \end{array}$$

hold for unary queries over unranked trees.

Part *b)* can be easily obtained from results in [35] which were shown by extending the proof of Kamp's theorem to trees. We give composition proofs of both results. This time we use the fact that the MSO (or FO) type of (T, s) is determined by the type of T_s and the type of the envelope $(\bar{T}_s, s)$ of s in T, where $\bar{T}_s$ is T from which T_s was cut, except s itself [24, 32]. The type of the subtree T_s can be expressed by using results for the Boolean case. Without sibling ordering, to compute the type of the envelope we have to count (up to a threshold) number of elements realizing certain types of subtrees, which leads to the counting temporal logics with the past.

5 Sibling-invariant queries

The notion of invariance is one of the central themes in finite model theory, and shows up in XML applications when one deals with formalisms that do not depend on a particular order imposed on siblings of node (most often in the case of XML constraints). It is known that adding relations such as order gives logics additional power even with respect to invariant queries. The standard example is parity in MSO: if we only have unary predicates, MSO=FO and hence parity (the number of elements of the universe is even) is not expressible. However, with *any* order on the universe, parity is definable in MSO, by checking if there exists a set that contains the every odd-numbered element of the ordering, and does not contain the last element. In the context of, for example, XML DTDs, a production $a \to \big((b|c)(b|c)\big)^*$ ensures that there is an even number of children of each a-node, and they are all labeled b or c, but the exact sibling ordering is irrelevant.

The general setting of invariance is as follows. Suppose we have a vocabulary θ and a sentence φ of some logic $\mathcal{L}$ in vocabulary θ expanded with another relation symbol R. Let $\mathcal{C}$ be a class of possible interpretations for R. Then on θ-structures $\mathfrak{M}$ this sentence is *$\mathcal{C}$-invariant* if for every possible interpretations $R_1, R_2 \in \mathcal{C}$ of R on $\mathfrak{M}$, we have $(\mathfrak{M}, R_1) \models \varphi \;\Leftrightarrow\; (\mathfrak{M}, R_2) \models \varphi$. We shall write $(\mathcal{L}[\theta] + \mathcal{C})_{\rm inv}$ for this class of formulae. If $\mathcal{C}$ is a class of linear orderings, we write $(\mathcal{L}[\theta] + <)_{\rm inv}$.

For the case of sibling-invariance, we shall use, as before, relations $\prec_{\rm ch}$ and $\prec_{\rm sb}$ for MSO, and thus sibling-invariant queries are those in $(\mathrm{MSO}[\prec_{\rm ch}] + \prec_{\rm sb})_{\rm inv}$. One can easily show that $\mathrm{MSO}[\prec_{\rm ch}, \prec_{\rm sb}]$ defines a linear order (lexicographic ordering over $\mathbb{N}^*$) and thus:

$$(\mathrm{MSO}[\prec_{\rm ch}] + \prec_{\rm sb})_{\rm inv} = (\mathrm{MSO}[\prec_{\rm ch}] + <)_{\rm inv}.$$

Order-invariant MSO has been studied [7, 8] and completely characterized for trees. Define CMSO, *counting* MSO [7], as MSO extended with modulo quantifiers $Q_p x \varphi(x, \cdot)$ with the semantics $\mathfrak{M} \models Q_p x \varphi(x, \cdot)$ iff the cardinality of the set $\{a \mid \mathfrak{M} \models \varphi(a, \cdot)\}$ is congruent to 0 modulo p. Then, using graph grammars and an algebraic approach to recognizability, [8] proved that $\mathrm{CMSO} = (\mathrm{MSO}[\prec_{\rm ch}] + <)_{\rm inv}$ over trees.

Thus, a natural idea for capturing $(\mathrm{MSO}[\prec_{\rm ch}] + \prec_{\rm sb})_{\rm inv}$ by a temporal logic seems to be expanding C_μ with $\Diamond^{\mathrm{mod}\ p}(\prec_{\rm ch})\varphi$ which is true in (T, s) iff the number of children s' of s with $(T, s') \models \varphi$ is congruent to 0 modulo p. The problem is that $\Diamond^{\mathrm{mod}\ p}(\prec_{\rm ch})X$ is not monotone in X and thus cannot be used in least fixed points. Using inflationary fixed points is not a solution either because a modal calculus with inflationary semantics behaves very differently from the μ-calculus [10]. Thus, we shall use a different approach and impose a syntactic restriction that allows modulo quantifiers to be used in least fixed points, in the spirit of extensions with arithmetic from [36].

Let $S_{k,p}$, for $k, p \geq 0$, be the set $\{k + np \mid n \in \mathbb{N}\}$. For an alphabet $\Sigma = \{a_1, \ldots, a_r\}$, consider the Parikh map $\Pi : \Sigma^* \to \mathbb{N}^r$ where the ith component of $\Pi(w)$ is the number of occurrences of a_i in w.

We define $C_\mu^{\rm mod}$ as an extension of C_μ with new modalities $\Diamond^{\bar{S}}(\prec_{\rm ch})\bar{\varphi}$, where $\bar{S} = (S_1, \ldots, S_m)$ is a tuple of sets of the form $S_{k,p}$ (equivalently, a tuple of pairs of elements of $\mathbb{N}$), and $\bar{\varphi} = (\varphi_1, \ldots, \varphi_m)$ is a tuple of formulae. Then $(T, v, s) \models \Diamond^{\bar{S}}(\prec_{\rm ch})\bar{\varphi}$ if s has n children $s_1, \ldots, s_n$ and there exists a string w_s of length n in $\{1, \ldots, m\}^*$ such that:

- if the jth symbol of w_s is i, then $(T, v, s_j) \models \varphi_i$;
- $\Pi(w_s) \in S_1 \times \ldots \times S_m$.

Then we close the formulae of $C_\mu^{\rm mod}$ by Boolean combinations, and simultaneous least fixed points $\mu(X_1, \ldots, X_m)\ \varphi(X_1, \ldots, X_m)$ for X_i's occurring positively in φ. It is routine to verify that such formulae φ define operators monotone in all X_i's and hence least fixed points exist.

We call an UNTA $\mathcal{A}$ *invariant* if for every state q and alphabet symbol a, every permutation of a string in the regular language $\delta(q, a)$ is also in $\delta(q, a)$. Thus, a run of an invariant automaton does not depend on the $\prec_{\rm sb}$ relation.

Theorem 5.1 *Over unranked trees,*

$$(\mathrm{MSO}[\prec_{\rm ch}] + \prec_{\rm sb})_{\rm inv} \;=\; C_\mu^{\rm mod}[\prec_{\rm ch}].$$

Moreover, the class of unranked trees definable by $(\mathrm{MSO}(\prec_{\rm ch}) + \prec_{\rm sb})_{\rm inv}$ *is precisely the class of trees accepted by deterministic invariant automata, and the translations between the three formalisms are effective.*

Proof sketch. For the proof we need a lemma about regular languages closed under permutations that can be is easily shown by an MSO game argument[1]. Assume $\Sigma = \{a_1, \ldots, a_r\}$.

Lemma 5.2 *A regular language $L \subseteq \Sigma^*$ is closed under permutation iff there exist a finite family $\mathcal{S}$ of r-tuples of sets of the form $S_{k,p}$ such that $w \in L$ iff for some $(S_1, \ldots, S_r) \in \mathcal{S}$, we have $\Pi(w) \in S_1 \times \ldots \times S_r$.*

Moreover, it is easy to see that for a regular language decidability of closure under permutations is decidable, and the family $\mathcal{S}$ can be effectively constructed.

The main ingredient of the proof is showing that $(\mathrm{MSO}(\prec_{\rm ch}) + \prec_{\rm sb})_{\rm inv}$ properties are definable by invariant automata, since such automata can easily be coded in $C_\mu^{\rm mod}[\prec_{\rm ch}]$ and the translation of $C_\mu^{\rm mod}[\prec_{\rm ch}]$ back into invariant MSO is easy. This in turn relies on Lemma 5.2 and a simple observation that if an automaton has a transition $\delta(q, a)$ that defines a language not closed under permutation, then either such an automaton defines a class of trees which is not sibling-invariant, or $\delta(q, a)$ can be replaced by a regular expression that is invariant under permutations and the tree language accepted by the automaton does not change. □

[1] This easy result may have appeared in the literature but we were unable to find it.

Since it is straightforward to encode in CMSO runs of automata in which all languages $\delta(q, a)$ are invariant under permutations (by Lemma 5.2), we obtain an easy automata-theoretic proof of the following:

Corollary 5.3 (Courcelle [8]) *Over unranked trees,* $(\text{MSO}[\prec_{\text{ch}}]+<)_{\text{inv}} = \text{CMSO}[\prec_{\text{ch}}]$.

6 Remarks and conclusion

Complexity of model-checking The complexity of model checking for logics considered here is either known or easily derived from known results. The most expressive logic considered here is $\text{MSO}[\prec_{\text{ch}}, \prec_{\text{sb}}] = L_\mu^{\text{full}}[\prec_{\text{ch}}, \prec_{\text{sb}}] = L_\mu[\prec_{\text{fc}}, \prec_{\text{sb}}]$. Unranked trees considered as labeled transition systems over relations $\prec_{\text{fc}}$ and $\prec_{\text{sb}}$ are acyclic, and hence by [27] the complexity of model checking is $O(\|\varphi\|^2 \cdot \|T\|)$. Furthermore, our equivalence proof produces an alternation-free L_μ formula (it is known that L_μ equals alternation-free L_μ over all acyclic transition systems [27]) which further reduces the complexity to $O(\|\varphi\| \cdot \|T\|)$, matching the best complexity of monadic datalog [15].

Logics corresponding to FO are extensions of $\text{CTL}^\star$. Then, in general, the complexity could be worse than for L_μ because existing translations of $\text{CTL}^\star$ into L_μ exhibit exponential or even doubly exponential blowup in the worst case [9, 4]. One can easily show that for the counting version $\text{CTL}^\star_{\text{count}}$ the complexity matches that of $\text{CTL}^\star$ (that is, $2^{O(\|\varphi\|)} \cdot \|T\|$), and for the past version $\text{CTL}^\star_{\text{past}}$ the best known result gives a PSPACE bound in terms of $\|\varphi\| + \|T\|$ [22].

XML applications Temporal logics have nice algorithmic properties not only with respect to model checking but also satisfiability. We believe they can be useful in studying various satisfiability problems in the XML context, that often arise in connection with the interaction of XML DTDs and constraints (e.g., integrity constraints, or existence of paths satisfying given XPath formulae).

A problem that received attention recently is consistency of DTDs and XPath: that is, whether an XPath formula and a DTD are consistent [25, 13]. In particular, [25] gives an EXPTIME upper bound which is complemented by the matching lower bound in [13]. In fact, it is the upper bound that is harder in this case, but it can be derived rather easily if one appeals to the connection between MSO and the full μ-calculus. Both DTDs and XPath can be coded in the L_μ^{full} (using backward modalities for the ancestor and older sibling axes). Then one uses the result of [38] showing that satisfiability for L_μ^{full} is in EXPTIME. In fact, a formula that is satisfiable, is also satisfiable on a tree of branching degree linear in the size of the formula. One still needs to enforce a few conditions on such a tree which could be done by using alternating two-way tree automata which are known to be equivalent to L_μ^{full} [38]. It is our hope that the connections of this kind could be explored further in the XML context.

Concluding remarks Figure 1 summarizes the main equivalences between fragments of FO and MSO and temporal logics.

There are several ways to extend this work. Quite a bit is known about the succinctness of various logical formalisms for trees [17], and we would like to see how temporal logics compare to other formalisms. We also would like to get more mileage out of the connection with well-studied temporal logics in terms of XML applications, perhaps by extending the idea of coding into the full mu-calculus illustrated in the previous section, and using two-way alternating tree automata to get further upper bound results for more complex problems such as implication of XML constraints.

Acknowledgments We thank Wenfei Fan, David Janin, Maarten Marx, Thomas Schwentick, and Luc Segoufin for comments and pointers.

References

[1] L. Afanasiev, M. Franceschet, M. Marx, M. de Rijke. CTL model checking for processing simple XPath queries. In *TIME 2004*, pages 117–124.

[2] M. Benedikt, W. Fan, G. Kuper. Structural properties of XPath fragments. In *ICDT 2003*, pages 79–95.

[3] M. Benedikt, L. Segoufin. Regular tree languages definable in FO. *STACS 2005*, to appear.

[4] G. Bhat, R. Cleaveland. Efficient model checking via the equational μ-calculus. In *LICS 1996*, pages 304–312.

[5] M. Bojanczyk, I. Walukiewicz. Characterizing EF and EX tree logics. In *CONCUR 2004*, pages 131–145.

[6] L. Cardelli, G. Ghelli. A query language based on the ambient logic. In *ESOP 2001*, pages 1–22.

[7] B. Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Inf.&Comput.* 85 (1990), 12–75.

[8] B. Courcelle. The monadic second-order logic of graphs V: On closing the gap between definability and recognizability. *TCS* 80 (1991), 153–202.

[9] M. Dam. CTL* and ECTL* as fragments of the modal mu-calculus. *TCS* 126 (1994), 77–96.

Logic	Boolean queries		Unary queries	
	unordered trees	ordered trees	unordered trees	ordered trees
FO	$\mathrm{FO}[\prec_{\mathrm{ch}}^*]$ $= \mathrm{CTL}^\star_{\mathrm{count}}[\prec_{\mathrm{ch}}]$ (see [29])	$\mathrm{FO}[\prec_{\mathrm{ch}}^*, \prec_{\mathrm{sb}}^*]$ $= \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ $= \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}]$ (see Theorem 3.4 and [26])	$\mathrm{FO}[\prec_{\mathrm{ch}}^*]$ $= \mathrm{CTL}^\star_{\mathrm{count,\ past}}[\prec_{\mathrm{ch}}]$ (see [35] and Theorem 4.3, part b)	$\mathrm{FO}[\prec_{\mathrm{ch}}^*, \prec_{\mathrm{sb}}^*]$ $= \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ $= \mathrm{CTL}^\star_{\mathrm{past}}[\prec_{\mathrm{ch}} \cup \prec_{\mathrm{sb}}]$ (see [26] and Theorem 4.1, part b)
MSO	$\mathrm{MSO}[\prec_{\mathrm{ch}}]$ $= C_\mu[\prec_{\mathrm{ch}}]$ (see [40, 11, 19] and Theorem 3.1)	$\mathrm{MSO}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ $= L_\mu^{\mathrm{full}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ (see Theorem 3.3)	$\mathrm{MSO}[\prec_{\mathrm{ch}}]$ $= C_\mu^{\mathrm{full}}[\prec_{\mathrm{ch}}]$ (see Theorem 4.3, part a)	$\mathrm{MSO}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ $= L_\mu^{\mathrm{full}}[\prec_{\mathrm{ch}}, \prec_{\mathrm{sb}}]$ (see Theorem 4.1, part a)

Figure 1. Summary of main equivalences

[10] A. Dawar, E. Grädel, S. Kreutzer. Inflationary fixed points in modal logic. *ACM TOCL* 5 (2004), 282–315.

[11] A. Dawar, D. Janin. On the bisimulation invariant fragment of monadic Σ_1 in the finite. In *FSTTCS 2004*, pages 224–236.

[12] E. A. Emerson, C. Jutla. Tree automata, mu-calculus and determinacy. In *FOCS 1991*, pages 368–377.

[13] W. Fan, M. Benedikt, F. Geerts. XPath satisfiability in the presence of DTDs. In *PODS'05*.

[14] M. Frick, M. Grohe. The complexity of first-order and monadic second-order logic revisited. In *LICS 2002*, 215–224.

[15] G. Gottlob, C. Koch. Monadic datalog and the expressive power of languages for web information extraction. *J. ACM* 51 (2004), 74–113.

[16] G. Gottlob, C. Koch. Monadic queries over tree-structured data. In *LICS 2002*, pages 189–202.

[17] M. Grohe, N. Schweikardt. Comparing the succinctness of monadic query languages over finite trees. In *CSL 2003*, pages 226–240.

[18] T. Hafer, W. Thomas. Computation tree logic CTL* and path quantifiers in the monadic theory of the binary tree. *ICALP 1987*, pages 269–279.

[19] D. Janin, G. Lenzi. Relating levels of the mu-calculus hierarchy and levels of the monadic hierarchy. In *LICS 2001*, pages 347–356.

[20] D. Janin, I. Walukiewicz. On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic. In *CONCUR 1996*, pages 263–277.

[21] H.W. Kamp. *Tense Logic and the Theory of Linear Order.* Ph.D. Thesis, UCLA, 1968.

[22] O. Kupferman, A. Pnueli. Once and for all. In *LICS'95*, pages 25–35.

[23] L. Libkin, F. Neven. Logical definability and query languages over unranked trees. In *LICS 2003*, pages 178–187.

[24] J. Makowsky. Algorithmic aspects of the Feferman-Vaught Theorem. *Annals of Pure and Applied Logic*, 126 (2004), 159–213.

[25] M. Marx. XPath with conditional axis relations. In *EDBT 2004*, pages 477–494.

[26] M. Marx. Conditional XPath, the first order complete XPath dialect. In *PODS 2004*, pages 13–22.

[27] R. Mateescu. Local model-checking of modal mu-calculus on acyclic labeled transition systems. In *TACAS 2002*, pages 281–295.

[28] G. Miklau, D. Suciu. Containment and equivalence for a fragment of XPath. *J. ACM* 51 (2004), 2–45.

[29] F. Moller, A. Rabinovich. Counting on CTL*: on the expressive power of monadic path logic. *Information and Computation*, 184 (2003), 147-159.

[30] F. Neven, Th. Schwentick. Expressive and efficient pattern languages for tree-structured data. In *PODS 2000*, pages 145–156.

[31] F. Neven. Automata, logic, and XML. In *CSL 2002*, pages 2–26.

[32] F. Neven, Th. Schwentick. Query automata over finite trees. *Theor. Comput. Sci.* 275 (2002), 633–674.

[33] D. Niwinski. Fixed points vs. infinite generation. In *LICS 1988*, pages 402–409.

[34] A. Rabinovich. Expressive power of temporal logics. In *CONCUR 2002*, pages 57–75.

[35] B.-H. Schlingloff. Expressive completeness of temporal logic of trees. *Journal of Applied Non-Classical Logics* 2 (1992), 157–180.

[36] H. Seidl, Th. Schwentick, A. Muscholl, P. Habermehl. Counting in trees for free. In *ICALP 2004*, pages 1136–1149.

[37] J. van Benthem. *Modal Logic and Classical Logic.* Bibliopolis, 1983.

[38] M. Y. Vardi. Reasoning about the past with two-way automata. In *ICALP 1998*, pages 628–641.

[39] V. Vianu. A web Odyssey: from Codd to XML. In *PODS'01*, pages.

[40] I. Walukiewicz. Monadic second-order logic on tree-like structures. *TCS* 275 (2002), 311–346.