Lindström Quantifiers and Leaf Language Definability

Hans-Jörg Burtschick
Fachbereich Informatik
TU Berlin
Franklinstr. 28/29
D-10587 Berlin

Heribert Vollmer
Theoretische Informatik
Universität Würzburg
Am Exerzierplatz 3
D-97072 Würzburg

Abstract

We introduce second-order Lindström quantifiers and examine analogies to the concept of leaf language definability. The quantifier structure in a second-order sentence defining a language and the quantifier structure in a first-order sentence characterizing the appropriate leaf language correspond to one another. Under some assumptions, leaf language definability and definability with second-order Lindström quantifiers may be seen as equivalent. Along the way we tighten the best up to now known leaf language characterization of the classes of the polynomial time hierarchy and give a new model-theoretic characterization of PSPACE.

1 Introduction

Initiated by Fagin's seminal paper [Fag74] characterizing NP as the class of generalized spectra, there has been a long line of research in characterizing complexity classes using notions from finite model theory (see the recent textbook [EF95]). In classical complexity theory problems are classified by defining restrictions on different computational resources as e.g. space or time. Using in contrast to this the syntactic complexity of a defining formula it is possible to measure the *descriptive* complexity of problems, e.g. by considering the complexity of quantifier prefixes of prenex formulae. Besides the well-known existential and universal quantifiers recently the expressive power of a certain kind of generalized quantifiers, the so called *Lindström quantifiers*, has been examined [Lin66]; for an overview see Chapter 10 of [EF95]. An extensive overview over the literature can be found in [Got95].

In the field of computational complexity characterizations of complexity classes by so called *leaf languages* have been studied intensively. This approach was introduced by Bovet, Crescenzi, and Silvestri in 1992 [BCS92] and independently by Vereshchagin in [Ver93] as a unified approach to define complexity classes. Consider a polynomial-time nondeterministic Turing machine M that prints a value on every path of its computation on some given input x. These values are at the leaves of the (not necessarily fully balanced) binary tree defined by the nondeterministic choices of M on input x. An ordering of the tuples in the program of M determines, for each nondeterministic choice node of the tree, which child is "left" and which is "right". This in turn fixes a left-to-right ordering of all the leaves. We call the corresponding sequence of values the *leaf string* of M on input x. (The particular ordering of tuples in the program of M does not matter.) Given now a (leaf) language B, a complexity class $\mathrm{Leaf}^{\mathrm{P}}(B)$ is defined as follows: A set L is in $\mathrm{Leaf}^{\mathrm{P}}(B)$ if and only if there is a nondeterministic Turing machine that on input x produces a leaf string which z such that $z \in B$ if and only if $x \in L$.

Starting with the somewhat surprising result that the class PSPACE can already be characterized in this way using only regular leaf languages [HLS$^+$93], a lot of characterizations were given [JMT94, HVW95, HVW96, CMTV96], leading to a number of remarkable and unexpected normal forms for such classes as PSPACE (via bottleneck machines [HLS$^+$93]), PP (via machines where accepting and rejecting computation paths form two clusters [HVW96]), and #P (where all relativizable closure properties were identified [HVW95]). Even separations of circuit classes could be proved using leaf languages as a technical vehicle: Allender showed in [All96], building on previous work by [CMTV96], that $\mathrm{TC}^0 \subsetneq \mathrm{PP}$.

It turns out, as we show here, that both concepts—Lindström quantifiers and leaf language definability—are closely related: Leaf language defined complexity classes can be characterized by second-order Lindström quantifiers that are followed by first-order formulas, and vice versa for every formula of this form there is a leaf language and a nondeterministic polynomially time-bounded Turing machine that define the corresponding model class. Thus Lindström quantifiers and leaf language definability are two points of view of essentially the same concept. Relying on both views one might hope to get new insights into the expressive power of Lindström quantifiers on ordered finite structures and properties of leaf language defined complexity classes.

We give some results in this direction in this paper: The polynomial time hierarchy introduced by Stockmeyer in [Sto73] is one of the central concepts in complexity theory. Thus of course there have been efforts to give leaf language characterizations of this hierarchy. In [HLS$^+$93] it was shown that using AC^0 as leaf language class, one obtains the union of all classes of the polynomial hierarchy, and using the k-th level of the Brzozowski- (or dot-depth-) hierarchy [Str94] as leaf language class, the boolean closure of the class Σ_k^p is obtained. In [JMT94] it was proved that $\mathrm{Leaf}^{\mathrm{P}}(\Sigma_k\text{-LOGTIME}) = \Sigma_k^p$. Using first-order definable leaf languages we prove a tighter connection: We show that the k-th level from the polynomial time hierarchy can be characterized using leaf languages that can be defined using first-order Σ_k formulae. As another example of how a combination of model-theoretic and complexity theoretic arguments can lead to new results, we show how to use a theorem from [HLS$^+$93] to derive a new model-theoretic characterization of PSPACE.

These results are special cases of a general theorem which states that if a leaf language is defined by a formula in prenex normal form starting with one Lindström quantifier and followed by a sequence of universal and existential quantifiers, then the characterized complexity class can be defined in a model-theoretic way by a second-order formula with exactly the same quantifier structure. Thus we see that using methods from finite model theory we are able to tighten existing complexity theoretic results. We think that generally it is a good idea to specify leaf languages in a descriptive way. One purpose when considering leaf language definability is to separate the computational aspect in the definition of a class from the way the computation (in form of the computation tree constructed by a nondeterministic machine) is evaluated. This evaluation scheme should therefore be given in a way free of computational aspects whatsoever. This goal is achieved using finite model theory to specify leaf languages.

Other connections between leaf language definability and finite model theory are established in [Vei96]. He considers logical reducibilities [EF95] as e.g. first-order reducibility and projection reducibility to succinct representations of languages. This concept of succinct representations may be viewed as a special case of leaf language definability (see also [BL96]). However, the results achieved in his paper are incomparable to ours.

2 Preliminaries

Leaf languages were introduced and used to define complexity classes in [BCS92, Ver93, HLS$^+$93] (see also the recent textbook [Pap94]). Let M be a polynomial time-bounded nondeterministic Turing machine. Given an input x, such a machine produces on every path a symbol from some finite alphabet Γ. Let $\beta_M(x)$ be the string of the so produced symbols (based on the natural order of paths of the machine). Then for $B \subseteq \Gamma^*$, we define $\mathrm{Leaf}^M(B) =_{\mathrm{def}} \left\{ x \mid \beta_M(x) \in B \right\}$. The class $\mathrm{Leaf}^{\mathrm{P}}(B)$ is the class of all languages $\mathrm{Leaf}^M(B)$ for all nondeterministic polynomial-time machines M. Here, B is the so called *leaf language* defining the class $\mathrm{Leaf}^{\mathrm{P}}(B)$. If $\mathcal{C}$ is a class of languages, then we define $\mathrm{Leaf}^{\mathrm{P}}(\mathcal{C}) =_{\mathrm{def}} \bigcup_{B\in\mathcal{C}} \mathrm{Leaf}^{\mathrm{P}}(B)$.

As it turned out [JMT94, HVW96], it plays a crucial role whether in the above definition we allow arbitrary Turing machines or we require that the computation trees produced have a special shape. For the case of full binary trees we use the notation $\mathrm{FBTLeaf}^{\mathrm{P}}(\mathcal{C})$, see [Her95]. Note, that with respect to the produced leaf string only the nondeterministic branches, but not the length of the path is relevant. In the rest of the paper we will use nondeterministic Turing machines M having the following property: Let k be the smallest natural number such that M is n^k time-bounded. Then all computation-paths are exactly of length n^k. This may be achieved in an unique way using left branches, if the length of a path is smaller than n^k. For every polynomial time-bounded nondeterministic Turing machine M there exists a polynomial time-bounded nondeterministic Turing machine M' fulfilling the above property such that $\beta_M(x) = \beta_{M'}(x)$ for all inputs x. Thus, when write $\mathrm{Leaf}^{\mathrm{P}}(\mathcal{C})$ we use Turing machines as described above.

It can be seen that in a special case the distinction between full binary trees and the general case is unnecessary. For this, given a language $A \subseteq \Gamma^*$ for some alphabet Γ, and a letter $\mathbf{1} \notin \Gamma$, define $\mathrm{pad}_{\mathbf{1}}(w)$ to be the regular language $\mathbf{1}^* a_1 \mathbf{1}^* a_2 \mathbf{1}^* \cdots \mathbf{1}^* a_n \mathbf{1}^*$ for $w = a_1 a_2 \cdots a_n$; and let $\mathrm{pad}_{\mathbf{1}}(A) =_{\mathrm{def}} \bigcup_{w \in A} \mathrm{pad}_{\mathbf{1}}(w)$. Say that a language class $\mathcal{L}$ is closed under padding iff for $A \in \mathcal{L}$ also $\mathrm{pad}_{\mathbf{1}}(A) \in \mathcal{L}$. Now it can be seen that $\mathrm{Leaf}^{\mathrm{P}}(\mathcal{L}) = \mathrm{FBTLeaf}^{\mathrm{P}}(\mathcal{L})$ whenever $\mathcal{L}$ is closed under padding [JMT94]. We remark that generally all language classes defined by automata (finite automata, pushdown automata, Turing machines) are closed under padding. This also applies, as we will see later, to most of the subclasses of the regular languages we consider.

Later we will also allow different padding symbols. For this let Δ be an alphabet such that $\Delta \cap \Gamma = \emptyset$. Then $\mathrm{pad}_{\Delta}(w) \subseteq (\Gamma \cup \Delta)^*$ is the language $\mathrm{pad}_{\Delta}(w) =_{\mathrm{def}} \Delta^* a_1 \Delta^* a_2 \Delta^* \cdots \Delta^* a_n \Delta^*$ for $w = a_1 a_2 \cdots a_n$.

In this paper, we consider classes that are characterized via logically defined leaf languages. Especially, we deal with the following classes of formulae:

FO is the class of all first-order formulae, SO is the class of all second-order formulae. We use the notations Σ_k^0 and Π_k^0 (Σ_k^1 and Π_k^1) to denote the subclasses of FO (SO, resp.) consisting of prenex formulae in Σ_k form or Π_k form, see [EF95]. For a formula ϕ we denote the class of finite ordered structures that are models of ϕ by $\mathrm{Mod}(\phi)$. Since we are interested in characterizations of complexity classes, we consider in this paper only *ordered finite structures*, that is, structures over signatures that do not contain constant or function symbols but do contain a symbol $<$ for a linear order on the universe. We do not use the successor relation.

We use an encoding f of strings into ordered finite structures defined in the usual way [Fag74] as follows: Let A be a finite set of symbols and $l = \lceil \log(|A|) \rceil$. The images of F are ordered finite structures over the signature $\sigma_l = \langle <, P_1, \ldots, P_l \rangle$ with unary predicates $P_1, \ldots, P_l$. For a fixed enumeration of the symbols in A, let a_i be the i-th symbol in A. If the x-th element in the string $w \in A^*$ is a_i, then $P_j(x) = m$ in $f(w)$, iff the j-th bit in the binary encoding of i is m, for all $1 \leq j \leq l$ and $m \in \{0, 1\}$. Note, that f is one-one and if $|A|$ is a power of two, then f is a bijection. If L is a language, then we write $F(L)$ for $\{\, f(w) \mid w \in L \,\}$. If $\mathcal{C}$ is a class of languages, then $F(\mathcal{C}) =_{\text{def}} \{\, F(L) \mid L \in \mathcal{C} \,\}$.

Furthermore we sometimes have to encode finite ordered structures into strings. Since f is one-one, f^{-1} is a function, too. We define an encoding e from finite ordered structures into strings that includes f^{-1} but also extends to map relations of arity greater than one into strings. Analogously to the above we write $E(\text{Mod}(\phi))$ for the set $\{\, e(\mathcal{A}) \mid \mathcal{A} \models \phi \,\}$, and generally for a structure class $\mathcal{K}$ we write $E(\mathcal{K})$ for $\{\, e(\mathcal{A}) \mid \mathcal{A} \in \mathcal{K} \,\}$. (Following [Fag74] we choose to use the symbols E and F for our encodings though they are maybe not very intuitive.)

If $\mathcal{L}$ is a logic (as e.g. FO or SO) and $\mathcal{K}$ is a complexity class, then we say that $\mathcal{L}$ *captures* $\mathcal{K}$ iff every property over (encodings of) structures decidable within $\mathcal{K}$ is expressible by $\mathcal{L}$ sentences (i.e., $F(\mathcal{K}) \subseteq \text{Mod}(\mathcal{L})$), and on the other hand for every fixed $\mathcal{L}$ sentence ϕ, determining whether $\mathcal{A} \models \phi$ can be done in $\mathcal{K}$ (i.e., $E(\text{Mod}(\mathcal{L})) \subseteq \mathcal{K}$). As an abbreviation we will most of the time simply write $\mathcal{K} = \mathcal{L}$.

If $\mathcal{L}$ is a logic, then $\text{Leaf}^{\text{P}}(\mathcal{L})$ is the complexity class characterized by the leaf language consisting of all words such that the corresponding structure is a model of an $\mathcal{L}$ formula, in symbols: $A \in \text{Leaf}^{\text{P}}(\mathcal{L})$ if there exist a $\phi \in \mathcal{L}$ and a polynomial time-bounded nondeterministic Turing machine M such that for all x, $x \in A \iff F(\beta_M(x)) \models \phi$.

In a seminal paper Fagin [Fag74] connected the complexity of computations with the syntactic complexity of logical languages. He showed the following: Let M be a nondeterministic polynomial time Turing machine that recognizes a language over some alphabet A. Let us assume that on every input, M's computation tree is a full binary tree. Fagin constructed an existential second-order formula Φ_M that describes the computation of M. The formula Φ_M is of the form $\exists \vec{X} \varphi_M(\vec{X})$ where φ_M is a first-order formula that has no free first-order variables and that additionally contains the free second-order variables $\vec{X}$. The arities and the number of the second-order variables are determined by the construction, see [Fag74].

Theorem 2.1 [Fag74]. $\text{NP} = \Sigma^1_1$, *i.e.* $F(\text{NP}) \subseteq \text{Mod}(\Sigma^1_1)$ *and* $E(\text{Mod}(\Sigma^1_1)) \subseteq \text{NP}$.

Let us consider a nondeterministic Turing machine that is n^k time-bounded and that produces a full binary computation tree for all inputs. It is possible to represent a path in a computation tree of depth n^k using a string $\pi \in \{0, 1\}^{n^k}$. We enumerate the paths $(\pi_i)_{1\le i\le 2^{n^k}}$ using the lexicographical ordering on $\{0,1\}^{n^k}$. Analogously, we can enumerate the assignments $(X_i^{\mathcal{A}})_{1\le i\le 2^{n^k}}$ of a k-ary second-order variable X using the lexicographical ordering on characteristic sequences. In this paper we want to use a direct correspondence between the paths π_i in the computation tree and the assignments $X_i^{\mathcal{A}}$ of X:

Proposition 2.2. *Let M be a n^k time-bounded nondeterministic Turing machine as above. There exists a Σ_1^1 formula of the form $\exists X \exists \vec{Y} \phi_{\text{Path}}(X,\vec{Y}) \wedge \phi_{\text{accept}}(\vec{Y})$, such that for all inputs w of length n and for all $1 \le i \le 2^{n^k}$, we have that M accepts w on path π_i iff $(f(w), X_i^{f(w)}) \models \exists \vec{Y} \phi_{\text{Path}}(X,\vec{Y}) \wedge \phi_{\text{accept}}(\vec{Y})$. Here, $\phi_{\text{Path}}(X,\vec{Y})$ expresses that $\vec{Y}$ encodes a path corresponding to nondeterministic guesses X, and $\phi_{\text{accept}}(\vec{Y})$ expresses that the path encoded by $\vec{Y}$ is accepting.*

Proof. (Sketch.) We slightly modify the construction in [Fag74] by introducing a k-ary second-order predicate $X^{(k)}$ such that its assignment represents the nondeterministic choice for each step of the computation. That is, for each assignment $X^{\mathcal{A}}$ there exists exactly one assignment $\vec{Y}^{\mathcal{A}}$ such that $(\mathcal{A}, X^{\mathcal{A}}, \vec{Y}^{\mathcal{A}}) \models \phi_{\text{Path}}$. ❑

3 Lindström Quantifiers

In order to extend the limited capabilities of first-order predicate logic it has been tried to enhance its expressive power by adding different sorts of operators or quantifiers. One systematic and formally elegant approach to this is to consider so called *Lindström quantifiers*.

A first-order formula ϕ with k free variables defines for every structure $\mathcal{A}$ the relation $\phi^{\mathcal{A}} =_{\text{def}} \left\{ \vec{a} \in A^k \mid \mathcal{A} \models \phi(\vec{a}) \right\}$, see [EF95].

Now the idea underlying the definition of generalized quantifiers is the following: Consider the classical first-order existential quantifier applied to some quantifier-free formula ψ with free variable x, i.e., consider the formula $\exists x \psi(x)$. Given an ordered finite structure $\mathcal{A}$, $\mathcal{A} \models \exists x \psi(x)$ iff $\psi^{\mathcal{A}} \neq \emptyset$. Analogously we get $\mathcal{A} \models \forall x \psi(x)$ iff $\psi^{\mathcal{A}} = A$ (A is the universe of $\mathcal{A}$). [Str94] defines modular quantifiers as follows: $\mathcal{A} \models \exists^{(q,r)} x \psi(x)$ iff $|\psi^{\mathcal{A}}| \equiv r \pmod q$. Thus it is very natural to define generalized quantifiers by considering arbitrary conditions on relations defined by formulae.

Every class of structures $K \subseteq \text{Struct}(\sigma)$ over a signature $\sigma = \langle <, P_1, \ldots, P_s \rangle$ defines the first-order Lindström quantifier Q_K^0 as follows: Let $\phi_1, \ldots, \phi_s$ be first-order formulae over signature $\tau = \langle <, \tau' \rangle$ such that for $1 \le i \le s$ the number of free variables in ϕ_i is equal to the arity of P_i. Then $Q_K^0 \vec{x}_1, \ldots, \vec{x}_s\, [\phi_1(\vec{x}_1), \ldots, \phi_s(\vec{x}_s)]$ is a Q_K^0FO formula. If $\mathcal{A} \in \mathrm{Struct}(\tau)$, then

$$\mathcal{A} \models Q_K^0 \vec{x}_1, \ldots, \vec{x}_s\, [\phi_1(\vec{x}_1), \ldots, \phi_s(\vec{x}_s)] \quad \text{iff} \quad (A, <^A, \phi_1^{\mathcal{A}}, \ldots, \phi_s^{\mathcal{A}}) \in K.$$

We want to stress the following point: Since we only deal with ordered structures both $\mathcal{A}$ and the elements of K are ordered. In the just given equivalence $\mathcal{A}$ is related to structure $(A, <^A, \phi_1^{\mathcal{A}}, \ldots, \phi_s^{\mathcal{A}})$, i.e. a structure with the same universe. In the just given definition, we made the convention that here we assume the order of A in both cases to be identical. To be more explicit, we define the semantics of Q_K^0 as follows: If $\mathcal{A}$ is an ordered structure $\mathcal{A} = (\mathcal{A}', <^A)$, then $(\mathcal{A}', <^A) \models Q_K^0 \vec{x}_1, \ldots, \vec{x}_s\, [\phi_1(\vec{x}_1), \ldots, \phi_s(\vec{x}_s)]$ iff $(A, <^A, \phi_1^{\mathcal{A}}, \ldots, \phi_s^{\mathcal{A}}) \in K$.

Following [Got95] we also write Q_K^+ for Q_K^1FO. Moreover let $Q_K^1\Sigma_k^0$ ($Q_K^1\Pi_k^0$) denote the subclasses where all the formulae $\phi_1, \ldots, \phi_s$ are prenex Σ_k (Π_k) formulae.

The just given definition is the original definition given by Lindström [Lin66], which the reader will also find in textbooks [Ebb85, EF95] and formulated with slight variations but equivalently in [Got95]. For our examinations, the following formulation will be useful (observe that this only makes sense for ordered structures):

Given a first-order formula ϕ with k free variables and a corresponding finite ordered structure $\mathcal{A}$, this defines the binary string $\chi_{\phi^{\mathcal{A}}}$ of length n^k ($n = |A|$), the so called *characteristic string* of $\phi^{\mathcal{A}}$. Now given a sequence $\phi_1, \ldots, \phi_s$ of formulae with k free variables each and a structure $\mathcal{A}$, we similarly get the tuple $(\chi_{\phi_1^{\mathcal{A}}}, \ldots, \chi_{\phi_s^{\mathcal{A}}})$, where $|\chi_{\phi_1^{\mathcal{A}}}| = \cdots = |\chi_{\phi_s^{\mathcal{A}}}| = n^k$. Certainly, there is a one-one correspondence between such tuples and strings of length n^k over a larger alphabet (in our case with 2^s elements) as follows. Let A_s be such an alphabet. Fix an arbitrary enumeration of A_s, i.e. $A_s = \{a_0, a_1, \ldots, a_{2^s-1}\}$. Then $(\chi_{\phi_1^{\mathcal{A}}}, \ldots, \chi_{\phi_s^{\mathcal{A}}})$ corresponds to the string $b_1 b_2 \cdots b_{n^k}$, where for $1 \le i \le n^k$, $b_i \in A_s$, $b_i = a_k$ for that k whose length s binary representation (possibly with leading zeroes) is given by $\chi_{\phi_1^{\mathcal{A}}}[i] \cdots \chi_{\phi_s^{\mathcal{A}}}[i]$. In symbols: $t_{A_s}(\chi_{\phi_1^{\mathcal{A}}}, \ldots, \chi_{\phi_s^{\mathcal{A}}}) = b_1 b_2 \cdots b_{n^k}$.

This leads us to the following definition: A sequence $[\phi_1, \ldots, \phi_s]$ is in *first-order word normal form,* iff the ϕ_i have the same number k of free variables. Next we show that if we restrict Q_K^+ such that the FO part is required to be in first-order word normal form we get no loss in expressive power.

Lemma 3.1. *Let* $Q_K^0 \vec{x}_1, \ldots, \vec{x}_s\, [\phi_1(\vec{x}_1), \ldots, \phi_s(\vec{x}_s)]$ *be a* Q_K^0FO *formula over signature* τ. *Let* $K \subseteq \mathrm{Struct}(\sigma)$ *for arbitrary* $\sigma = \langle <, P_1, \ldots, P_s \rangle$. *Then there are a signature* σ', *a class of structures* $K' \subseteq \mathrm{Struct}(\sigma')$, *and first-order formulae* $[\phi_1', \ldots, \phi_s']$ *over* τ *in word normal form such that* $\mathrm{Mod}(Q_K^0 \vec{x}_1, \ldots, \vec{x}_s\, [\phi_1(\vec{x}_1), \ldots, \phi_s(\vec{x}_s)]) = \mathrm{Mod}(Q_{K'}^0 \vec{y}\, [\phi_1'(\vec{y}), \ldots, \phi_s'(\vec{y})])$.

Proof. Let k be the maximal arity of predicate symbols P_i ($1 \leq i \leq s$) in σ. Define σ' to consist of $<$ for a linear order plus s predicates of arity k each. Define K' to consist of exactly those structures $\mathcal{A}' = (A, <, R'_1, \ldots, R'_s)$ for which there is an $\mathcal{A} = (A, <, R_1, \ldots, R_s) \in K$ such that for all $1 \leq i \leq s$ we have: $(a_1, \ldots, a_{\ell_i}) \in R_i \iff (0, \ldots, 0, a_1, \ldots, a_{\ell_i}) \in R'_i$ (where ℓ_i is the arity of P_i, and 0 is a symbol for the minimal element in A). For the characteristic strings this has the consequence that $\chi_{R'_i} = \chi_{R_i} 0^{|A|^k - |A|^{\ell_i}}$. Now it is not too hard to build the formulae $\phi'_i, \ldots, \phi'_s$ such that for all $1 \leq i \leq s$, $(a_1, \ldots, a_{\ell_i}) \in \phi_i^{\mathcal{A}} \iff (0, \ldots, 0, a_1, \ldots, a_{\ell_i}) \in (\phi'_i)^{\mathcal{A}}$. ❑

Thus the following definition of *Lindström quantifiers over languages* is equivalent (w.r.t. expressive power) to the original definition of quantifiers over arbitrary structure classes K:

Let $[\phi_1, \ldots, \phi_s]$ be in first-order word normal form. Let Γ be an alphabet such that $|\Gamma| \leq 2^s$, and let $B \subseteq \Gamma^*$. Then $\mathcal{A} \models Q_B^0 \vec{x}\, [\phi_1(\vec{x}), \ldots, \phi_s(\vec{x})]$ iff $t_\Gamma(\chi_{\phi_1^{\mathcal{A}}}, \ldots, \chi_{\phi_s^{\mathcal{A}}}) \in B$.

We remark that our definition of Lindström quantifiers over languages resembles very much the definition of *group quantifiers* from [BIS90]. If G is the word problem of some finite group, then Q_G in our notation is exactly the group quantifier given by G.

To characterize the power of leaf languages defined with Q_B^+ formulae, we now introduce second-order Lindström quantifiers.

Given a formula ϕ with free second-order variables $P_1, \ldots, P_m$ and a structure $\mathcal{A}$, define $\phi^{2^{\mathcal{A}}} =_{\text{def}} \left\{ (R_1^{\mathcal{A}}, \ldots, R_m^{\mathcal{A}}) \;\middle|\; \mathcal{A} \models \phi(R_1^{\mathcal{A}}, \ldots, R_m^{\mathcal{A}}) \right\}$, and let $\chi_{\phi^{2^{\mathcal{A}}}}$ be the corresponding characteristic string, the order of vectors of relations being the natural one induced by the underlying order of the universe. If the arities of $P_1, \ldots, P_m$ are $r_1, \ldots, r_m$, resp., then the length of $\chi_{\phi^{2^{\mathcal{A}}}}$ is $2^{n^{r_1} + \cdots + n^{r_m}}$.

Let $\sigma = \langle <, \sigma_1, \ldots, \sigma_s \rangle$ be a signature, where $\sigma_i = \langle P_{i,1}, \ldots, P_{i,m_i} \rangle$ for $1 \leq i \leq s$. Thus σ is a signature consisting of a linear order plus a sequence of s signatures with only predicate symbols each. Let $\ell_{i,j}$ be the arity of $P_{i,j}$. A *second-order structure* of signature σ is a tuple $\mathcal{A} = (A, <^A, \mathcal{R}_1, \ldots, \mathcal{R}_s)$, where for every $1 \leq i \leq s$, $\mathcal{R}_i \subseteq \left\{ (R_{i,1}, \ldots, R_{i,m_i}) \;\middle|\; R_{i,j} \subseteq A^{\ell_{i,j}} \right\}$. Given now a signature $\tau = \langle <, \tau' \rangle$ and second-order formulae $\phi_1(\vec{X}_1), \ldots, \phi_s(\vec{X}_s)$ over τ where for every $1 \leq i \leq s$ the number and arity of free predicates in ϕ_i corresponds to σ_i. Let $\mathcal{K}$ be a class of second-order structures over σ. Then $Q_{\mathcal{K}}^1 \vec{X}_1, \ldots, \vec{X}_s \left[\phi_1(\vec{X}_1), \ldots, \phi_s(\vec{X}_s) \right]$ is a $Q_{\mathcal{K}}^1$SO formula. If $\mathcal{A} \in \text{Struct}(\tau)$, then

$$\mathcal{A} \models Q_{\mathcal{K}}^1 \vec{X}_1, \ldots, \vec{X}_s \left[\phi_1(\vec{X}_1), \ldots, \phi_s(\vec{X}_s) \right] \quad \text{iff} \quad (A, <^A, \phi_1^{2^{\mathcal{A}}}, \ldots, \phi_s^{2^{\mathcal{A}}}) \in \mathcal{K}.$$

Again we would like to stress that we assume the order on A to be the same on both sides of the equivalence.

Again, we want to talk about second-order Lindström quantifiers defined by languages. Thus we define analogously to the above: A sequence $\left[\phi_1(\vec{X}_1), \dots, \phi_s(\vec{X}_s)\right]$ of second-order formulae is in *second-order word normal form*, if the $\phi_1, \dots, \phi_s$ have the same predicate symbols, i.e. in the above terminology $\sigma_1 = \cdots = \sigma_s = \langle P_1, \dots, P_m \rangle$. Let for $1 \leq i \leq m$ the arity of P_i be r_i. Observe that in this case, $|\chi_{\phi_1^{2\mathcal{A}}}| = \cdots = |\chi_{\phi_s^{2\mathcal{A}}}| = 2^{n^{r_1}+\cdots+n^{r_m}}$ (for $n = |A|$), thus $(\chi_{\phi_1^{2\mathcal{A}}}, \dots, \chi_{\phi_s^{2\mathcal{A}}})$ corresponds to a word of the same length over an alphabet of cardinality 2^s. As before, we can assume second-order word normal form without loss of generality:

Lemma 3.2. *Let* $Q^1_{\mathcal{K}}\vec{X}_1, \dots, \vec{X}_s\left[\phi_1(\vec{X}_1), \dots, \phi_s(\vec{X}_s)\right]$ *be a* $Q^1_{\mathcal{K}}\mathrm{SO}$ *formula over signature* τ. *Let* $\mathcal{K}$ *be a class of second-order structures over some signature* $\sigma = \langle <, \sigma_1, \dots, \sigma_s \rangle$ *where* $\sigma_i = \langle P_{i,1}, \dots, P_{i,m_i} \rangle$ *for* $1 \leq i \leq s$. *Then there are a second-order structure* $\mathcal{K}'$ *over a suitable signature* σ' *and formulae* $[\phi'_i, \dots, \phi'_s]$ *over* τ *in second-order word normal form such that* $\mathrm{Mod}(Q^1_{\mathcal{K}}\vec{X}_1, \dots, \vec{X}_s\left[\phi_1(\vec{X}_1), \dots, \phi_s(\vec{X}_s)\right]) = \mathrm{Mod}(Q^1_{\mathcal{K}'}\vec{Y}\left[\phi'_1(\vec{Y}), \dots, \phi'_s(\vec{Y})\right])$.

Proof. The proof is similar to the proof of Lemma 3.1. Define $\sigma' = \langle <, \sigma'_1, \dots, \sigma'_s \rangle$ such that for $1 \leq i \leq s$ each σ'_i contains m predicate symbols, where m is the maximal number of predicate symbols in any σ_j, $1 \leq j \leq s$. The ℓth predicate symbol in any σ'_i is of arity k_ℓ, where k_ℓ is the maximal arity of all $\ell$the predicate symbols in any σ_j, $1 \leq j \leq s$. $\mathcal{K}'$ consists of those second-order structures $\mathcal{A} = (A, <^A, \mathcal{R}'_1, \dots, \mathcal{R}'_s)$ for which there is an $\mathcal{A} = (A, <^A, \mathcal{R}_1, \dots, \mathcal{R}_s)$ in $\mathcal{K}$ such that for $1 \leq i \leq s$ we have $\mathcal{R}'_i =_{\text{def}} \left\{ (R'_{i,1}, \dots, R'_{i,m_i}, \emptyset, \dots, \emptyset) \;\middle|\; (R_{i,1}, \dots, R_{i,m_i}) \in \mathcal{R}_i \right\}$. Here m_i is the number of predicate symbols in σ_i, and $R'_{i,j}$ is obtained from $R_{i,j}$ as in Lemma 3.1. Now it is a not too hard exercise to define formulae $\phi'_i, \dots, \phi'_s$ such that the following holds: $(A, <^A, \phi_1^{2\mathcal{A}}, \dots, \phi_s^{2\mathcal{A}}) \in \mathcal{K} \iff (A, <^A, (\phi'_1)^{2\mathcal{A}}, \dots, (\phi'_s)^{2\mathcal{A}}) \in \mathcal{K}'$ which implies the statement of the lemma. ❑

Let $\left[\phi_1(\vec{X}_1), \dots, \phi_s(\vec{X}_s)\right]$ be a sequence of second-order formulae in second-order word normal form as above. Given now a language $B \subseteq \Gamma^*$ with $|\Gamma| \leq 2^s$, the second-order Lindström quantifier given by B is defined by $\mathcal{A} \models Q^1_B\vec{X}\left[\phi_1(\vec{X}_1, \dots, \phi_s(\vec{X}_s)\right]$ iff $t_\Gamma(\chi_{\phi_1^{2\mathcal{A}}}, \dots, \chi_{\phi_s^{2\mathcal{A}}}) \in B$. Lemma 3.2 shows that second-order Lindström quantifiers given by languages are equivalent (w.r.t. expressibility power) to those given by arbitrary second-order structures.

Similarly to the above, $Q^1_B\Sigma^1_k$ ($Q^1_B\Pi^1_k$) denotes those subclasses of $Q^1_B\mathrm{SO}$, where all the formulae $\phi_1, \dots, \phi_s$ are prenex Σ^1_k (Π^1_k) formulae. For a class of languages $\mathcal{C}$ we use the notation $Q_{\mathcal{C}}$ with the obvious meaning, e.g. $Q^0_{\mathcal{C}}\mathrm{FO}$ denotes the union of all $Q^0_B\mathrm{FO}$ over all $B \in \mathcal{C}$.

4 Padding

In the sequel we will use logically defined leaf languages. As already pointed out in the preliminaries, padding sometimes plays a crucial role in the leaf context. Thus we will consider besides padding on languages defined earlier also padding on model classes. From the definition of logically defined leaf languages in Sect. 2 it is clear that we are mainly interested in structures which are images of the f encoding.

Let the encoding $t_s\colon (\{0,1\}^*)^s \to (\{0,1\}^s)^*$ be such that $t_s(w_1,\dots,w_s) = (w_{1,1},\dots,w_{s,1})\cdots(w_{1,n},\dots,w_{s,n})$, where $|w_1| = \cdots = |w_s| = n$ and $w_{i,j}$ is the jth symbol of w_i for $1 \le i \le s$, $1 \le j \le n$.

Let $\sigma = \langle <, P_1, \dots, P_s\rangle$, where all predicate symbols P_i ($1 \le i \le s$) are of the same arity. For a structure $\mathcal{A} = (A, R_1, \dots, R_s) \in \mathrm{Struct}(\sigma)$ let $t_s(\mathcal{A}) =_{\mathrm{def}} t_s(\chi_{R_1}, \dots, \chi_{R_s})$.

The idea behind padding on structures is that we want to insert a padding symbol in a tuple of characteristic strings, i.e. in $t_s(\mathcal{A})$. For this end, we have to extend our alphabet $\{0,1\}^s$ by a padding symbol. Actually, what we do is we take as new alphabet $\{0,1\}^{s+1}$ and all symbols in $\Delta =_{\mathrm{def}} \{\, (1, a_1, \dots, a_s) \mid a_1, \dots, a_s \in \{0,1\} \,\}$ may be used as padding symbols. More formally:

Let $\sigma_s =_{\mathrm{def}} \langle <, P_p, P_1, \dots, P_s\rangle$, where P_p is of the same arity as the other predicate symbols. Define $\mathrm{pad}(\mathcal{A}) \subseteq \mathrm{Struct}(\sigma')$ as follows: $\mathrm{pad}(\mathcal{A}) =_{\mathrm{def}} \{\, \mathcal{A}' \mid t_{s+1}(\mathcal{A}') \in \mathrm{pad}_\Delta(t_s(\mathcal{A})) \,\}$. (Here pad_Δ refers to padding on strings as defined in Sect. 2.) If K is a model class, then $\mathrm{pad}(K) =_{\mathrm{def}} \bigcup_{\mathcal{A} \in K} \mathrm{pad}(\mathcal{A})$. If $\mathcal{K}$ is a class of model classes, then $\mathrm{pad}(\mathcal{K}) =_{\mathrm{def}} \{\, \mathrm{pad}(K) \mid K \in \mathcal{K} \,\}$.

We say a logic $\mathcal{L}$ is closed under padding iff the corresponding class of model classes is, i.e. if $\mathrm{pad}(\mathrm{Mod}(\mathcal{L})) \subseteq \mathrm{Mod}(\mathcal{L})$.

In [Bur96] some logics closed under padding and some not closed under padding are identified. The in our context relevant results are the following.

Proposition 4.1. *1. For $k \ge 1$, Σ^0_k and Π^0_k are closed under padding.*

2. If $\mathcal{C}$ is closed under padding, then $Q^0_{\mathcal{C}}\Sigma^0_k$ and $Q^0_{\mathcal{C}}\Pi^0_k$ are closed under padding for all $k \ge 0$.

Proof. (Sketch.) The principal idea is that one has to relativize the occurring quantifiers, e.g. instead of "for all positions n" we now have "for all position n such that the nth letter is not a padding symbol." This certainly does not affect the linear order. Concerning the other relations, the such obtained formula does exactly what is required, i.e. it models the padded version of the original model. More details can be found in [Bur96]. ❑

5 Main Results

Before we come to our main result we examine leaf language characterizations of the classes of the polynomial time hierarchy.

Lemma 5.1. $\mathrm{Leaf}^{\mathrm{P}}(\Sigma_k^0) \subseteq \Sigma_k^1$ *for all* $k \geq 1$.

In [JMT94], $\mathrm{Leaf}^{\mathrm{P}}(\Sigma_k\text{-LOGTIME}) = \Sigma_k^p$ was proved. Thus, since $\Sigma_k^0 \subseteq \Sigma_k$-LOGTIME and $\Sigma_k^1 = \Sigma_k^p$ [Sto73], our lemma follows, but we give a new proof here, since we want to generalize it later.

Proof. For a given Σ_k^0-formula ϕ_{FO} and a polynomially time-bounded nondeterministic Turing machine M, we construct a Σ_k^1 formula ϕ_{SO} and then show for all $w \in \Sigma^*$ that

$$f(w) \models \phi_{SO} \text{ iff } w \in \mathrm{Leaf}^{\mathrm{P}}(\phi_{FO}).$$

Let M be a polynomially time-bounded nondeterministic Turing machine. Let us first consider the case, that Γ, namely the set of symbols that M produces on every computation-path (see Sect. 2), has two elements. We will discuss the other case at the end of the construction.

Let $\phi_M \equiv \exists X \exists \vec{Y}\ \phi_{\mathrm{Path}}(X, \vec{Y}) \wedge \phi_{\mathrm{accept}}(\vec{Y})$ be the Σ_1^1-formula which formalizes the computation of M as described in Sect. 2, Proposition 2.2. An assignment of X represents the nondeterministic choices. For a fixed assignment of X there exists exactly one assignment of $\vec{Y}$ that represents a computation path of M. For all suitable finite ordered structures $\mathcal{A}$ the formula $\phi_{\mathrm{Path}}(X, \vec{Y})$ is true on $(\mathcal{A}, X^{\mathcal{A}}, \vec{Y}^{\mathcal{A}})$, iff $\vec{Y}^{\mathcal{A}}$ represents a correct computation path w. r. t. the nondeterministic choices given by $X^{\mathcal{A}}$.

Let ϕ_{FO} be the Σ_k^0-formula over $(P^{(1)}, <)$ of the form

$$\exists \vec{x}_1 \forall \vec{x}_2 \ldots Q_k \vec{x}_k \varphi_{FO}(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_k).$$

We construct a Σ_k^1-formula ϕ_{SO} by replacing the variables and atomic predicates in φ_{FO} in the following way:

[$\exists x_i$] We replace all first-order variables x_i in φ_{FO} for $i = k$ down to $i = 1$ by tuples of second-order variables $(X_i, \vec{Y}_i)$ and thus transform the existential first-order quantifier that binds x_i into a sequence of second-order quantifiers that bind X_i and $\vec{Y}_i$. To ensure that an assignment of $\vec{Y}$ represents a correct computation path, we replace $\exists x_i \varphi_i$ by $\exists X_i \exists \vec{Y}_i \phi_{\mathrm{Path}}(X_i, \vec{Y}_i) \wedge \varphi_i$. In order to obtain a formula in prenex normal form, we rename the variables in ϕ_{Path} such that ϕ_{Path} and φ_i have no variables in common.

$[\forall x_i]$ We replace the universal quantifier $\forall x_i \varphi_i$ by $\forall X_i \forall \vec{Y}_i \phi_{\text{Path}}(X_i, \vec{Y}_i) \rightarrow \varphi_i$. Again we rename the variables in ϕ_{Path} such that ϕ_{Path} and φ_i have no variables in common.

$[P(x_i)]$ We replace each occurrence of $P(x_i)$ by the Σ^1_k-formula $\phi_{\text{accept},i}(\vec{Y}_i)$, which we obtain from $\phi_{\text{accept}}(\vec{Y})$ by a suitable renaming of the variables.

$[(x_i < x_j)]$ We have to extend the ordering that is given in the ordered structure to assignments of $X, \vec{Y}$. This can be achieved by well-known techniques.

As described in Sect. 2, it is possible to encode the symbols of Γ in binary. For $|\Gamma| > 2$, we have to consider first-order formulae over signatures that have $m = \lceil \log |\Gamma| \rceil$ unary predicates $P_1, \ldots, P_m$. We replace each $P_j(x_i)$ by a formula $\phi_{j,i}(\vec{Y}_i)$. For each $c \in \Gamma$ we obtain from $\phi_{\text{accept},i}(\vec{Y}_i)$ a formula $\varphi^c_i(\vec{Y}_i)$ which is true, iff M produces c on the path represented path. The formula $\phi_{j,i}(\vec{Y}_i)$ is a conjunction of the formulae $\varphi^c_i(\vec{Y}_i)$ for those $c \in \Gamma$, such that the j-th bit in the binary encoding of c is 1.

To see that $w \in \text{Leaf}^{\text{P}}(\phi_{\text{FO}})$, iff $f(w) \in \text{Mod}(\phi_{\text{SO}})$, for all $w \in \Sigma^*$, observe that there is a one-one correspondence between the elements in the universe of $f(\beta_M(w))$ (the positions of the symbols in the leaf string) and the assignments of $X^{f(w)}$.

❑

Lemma 5.2. $\Sigma^1_k \subseteq \text{Leaf}^{\text{P}}(\Sigma^0_k)$ *for all* $k \geq 1$.

Since even DLOGTIME $\not\subseteq$ FO, this lemma (in contrast to Lemma 5.1) does not follow from $\text{Leaf}^{\text{P}}(\Sigma_k\text{-LOGTIME}) = \Sigma^p_k$ proved in [JMT94], but in fact strengthens their result.

Proof. Let ϕ_{SO} be a Σ^1_k formula with m second-order variables $S_1, \ldots, S_m$. We construct a polynomially time-bounded NTM M and a Σ^0_k formula ϕ_{FO} defining a leaf-language $F^{-1}(\text{Mod}(\phi_{\text{FO}})) \subseteq \Gamma^*$, such that for all ordered finite structures $\mathcal{A}$ over the corresponding signature: $\mathcal{A} \in \text{Mod}(\phi_{\text{SO}})$ iff $e(\mathcal{A}) \in \text{Leaf}^{\text{P}}(\phi_{\text{FO}})$

On input $e(\mathcal{A})$ the NTM M determines nondeterministically the assignments of the second-order variables from S_1 to S_m. That is, on each computation path, M writes the characteristic sequence of one possible assignment on its work tapes. Using these characteristic sequences, M evaluates the first-order part of ϕ_{SO}. We use two leaf symbols 0 and 1 to denote the result of the evaluation.

For Σ^1_1 formulas, the corresponding leaf language is defined by $\exists x P(x)$, and for Π^1_1 formulas the corresponding leaf language is defined by $\forall x P(x)$.

For $k > 1$, we have to mark for all second-order variables S_i ($1 \leq i \leq m$) the paths that share the same assignment of $S^{\mathcal{A}}_i$ as shown in Fig. 1.

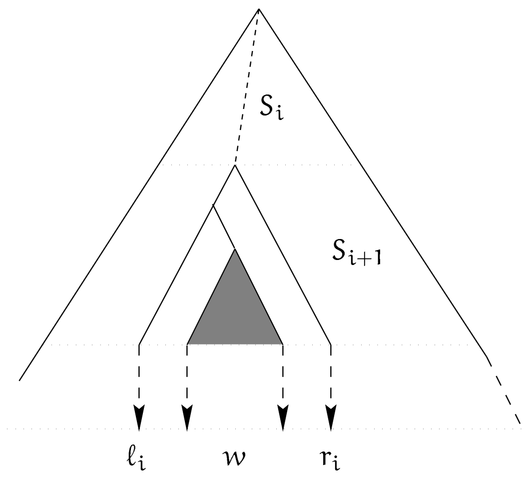

Figure 1: Construction of machine M

We introduce $2 \cdot (m-1)$ new leaf symbols $\ell_1, \ldots, \ell_{m-1}$ and $r_1, \ldots, r_{m-1}$. The symbols of of the leaf string w between ℓ_i and r_i correspond to all those paths that share the same assignment for S_i.

As described in the preliminaries, the symbols of Γ can be encoded using $\lceil \log |\Gamma| \rceil$ unary predicates. By boolean combinations of these, we construct formulae that express that a path produces 0, 1, ℓ_i, or r_i for $1 \leq i \leq m-1$. We abbreviate these formulae by φ_0, φ_1, $\varphi_{\ell,i}$, and $\varphi_{r,i}$ resp. The formula $\neg(\varphi_0(x) \vee \varphi_1(x))$ will be abbreviated by $\varphi_{\text{marked}}(x)$.

For $1 \leq i < m$ let $\text{Region}_i(x, y)$ be an abbreviation for the formula

$$(x < y) \wedge \varphi_{\ell,i}(x) \wedge \varphi_{r,i}(y) \wedge \forall z_i (x \leq z_i \leq y) \to \neg(\varphi_{\ell,i}(z_i) \vee \varphi_{r,i}(z_i)).$$

For $k > 1$ we construct inductively a sequence of formulae $\phi_k, \ldots, \phi_2, \phi_{\text{FO}}$. Let $\phi_k(x_{\text{left},k-1}, x_{\text{right},k-1})$ be the formula

$$\exists x_k (x_{\text{left},k-1} \leq x_k \leq x_{\text{right},k-1}) \wedge \varphi_1(x_k),$$

if the rightmost quantifier is existential and

$$\forall x_k (x_{\text{left},k-1} \leq x_k \leq x_{\text{left},k-1} \wedge \neg\varphi_{\text{marked}}(x_k)) \to \varphi_1(x_k),$$

otherwise. For the i-th quantifier block $(2 < i < k-1)$ let us first consider the case, that the quantifier block itself consists of existential quantifiers and that the quantifier block to the right consists of universal quantifiers. Let S_j be the rightmost variable in the i-th quantifier block. We define $\phi_i(x_{\text{left},i-1}, x_{\text{right},i-1})$ to be the formula

$$\begin{aligned}\exists x_{\text{left},i} \exists x_{\text{right},i} \big[x_{\text{left},i-1} \leq x_{\text{left},i} < x_{\text{right},i} \leq x_{\text{right},i-1} \wedge \text{Region}_j(x_{\text{left},i}, x_{\text{right},i}) \big] \\ \wedge\, \phi_{i+1}(x_{\text{left},i}, x_{\text{right},i}).\end{aligned}$$

Transforming this into prenex normal form we get a quantifier prefix starting with $\exists x_{\text{left},i} \exists x_{\text{right},i} \forall z_j \forall x_{\text{left},i+1} \forall x_{\text{right},i+1} \dots$. Let us now consider the case, that the i-th quantifier block $(1 < i < k-1)$ consists of universal quantifiers and that the quantifier blocks to the left and to the right consist of existential quantifiers. Let S_j be the rightmost variable in the i-th quantifier block. We define $\phi_i(x_{\text{left},i-1}, x_{\text{right},i-1})$ to be the formula

$$\begin{aligned}\exists x_{\text{left},i} \exists x_{\text{right},i} \big[x_{\text{left},i-1} \leq x_{\text{left},i} < x_{\text{right},i} \leq x_{\text{right},i-1} \land \text{Region}_j(x_{\text{left},i}, x_{\text{right},i})\big] \\ \to \phi_{i+1}(x_{\text{left},i}, x_{\text{right},i}).\end{aligned}$$

Transforming this into prenex normal form we get a quantifier prefix starting with $\forall x_{\text{left},i} \forall x_{\text{right},i} \exists z_j \exists x_{\text{left},i+1} \exists x_{\text{right},i+1} \dots$.

Let S_j be the rightmost variable in the first quantifier block. If S_1 is existentially quantified then let ϕ_{FO} be the formula

$$\exists x_{\text{left},1} \exists x_{\text{right},1} \ \text{Region}_j(x_{\text{left},1}, x_{\text{right},1}) \land \phi_2(x_{\text{left},1}, x_{\text{right},1}),$$

otherwise let ϕ_{FO} be

$$\forall x_{\text{left},1} \forall x_{\text{right},1} \ \text{Region}_j(x_{\text{left},1}, x_{\text{right},1}) \to \phi_2(x_{\text{left},1}, x_{\text{right},1}).$$

❑

From Lemma 5.1 and Lemma 5.2 we get the following leaf language characterization of the classes of the polynomial time hierarchy:

Theorem 5.3. $\text{Leaf}^P(\Sigma_k^0) = \Sigma_k^1$ *for all* $k \geq 1$.

A close inspection of the proof in [Fag74] shows that there is a direct correspondence between computation-paths and assignments of the existentially quantified second-order variables. Thus, using *second-order* Lindström quantifiers we can describe the concept of leaf language definability in terms of model-theoretic notions. Note, that we do *not* require the computation tree to be a full binary tree.

Proposition 5.4. *Let* $B \subseteq \Gamma^*$ *be a language, let* s *be a number such that* $2^s \geq |\Gamma|$, *and let* M *be a nondeterministic polynomial time machine as described in Sect. 2 (i.e. with all paths of the same length). Then there exists a formula* $\exists \vec{Y} \phi_{\text{Path}}(X, \vec{Y})$ *and a sequence of formulae* $\phi_{\text{leaf},1}(\vec{Y}), \dots, \phi_{\text{leaf},s}(\vec{Y})$ *such that*

$$\text{Leaf}^M(B) = \text{Mod}\big(Q_B X\big[\exists \vec{Y} \phi_{\text{Path}}(X, \vec{Y}) \land \phi_{\text{leaf},1}(\vec{Y}), \dots, \exists \vec{Y} \phi_{\text{Path}}(X, \vec{Y}) \land \phi_{\text{leaf},s}(\vec{Y})\big]\big).$$

Proof. Let $\exists X \exists \vec{Y} \phi_{\text{Path}}(X, \vec{Y}) \wedge \phi_{\text{accept}}(\vec{Y})$ be the formula, which is described in Proposition 2.2. For each computation path of M on input x there exists exactly one assignment $X^{f(x)}$ such that

$$\left(f(x), X^{f(x)}\right) \quad \models \quad \exists \vec{Y} \phi_{\text{Path}}(\vec{Y}).$$

For $s = 1$ we modify the formula $\phi_{\text{accept}}(\vec{Y})$ such that it is true, iff M produces the leaf symbol c_1. We denote the resulting formula by $\phi_{\text{leaf},1}(\vec{Y})$. In the corresponding encoding of $\beta_M(x)$ the unary predicate is true, iff c_1 is produced. Thus, we have

$$L \in \text{Leaf}^M(B) \quad \text{iff} \quad L \in \text{Mod}\left(Q_B X\left[\exists \vec{Y} \phi_{\text{Path}}(X, \vec{Y}) \wedge \phi_{leaf}(\vec{Y})\right]\right).$$

For $s > 1$ we construct the formulae $\phi_{\text{leaf},1}(\vec{Y}), \ldots, \phi_{\text{leaf},s}(\vec{Y})$ from $\phi_{\text{accept}}(\vec{Y})$ according to the encoding t_Γ (see Sect. 3): For all $a \in \Gamma$ let ϕ_a be a formula that is true for $(f(x), X^{f(x)}, Y^{f(x)})$, iff M produces the leaf-symbol a on the path that corresponds to $\left(X^{f(x)}, Y^{f(x)}\right)$. Each formula $\phi_{\text{leaf},i}(\vec{Y})$ now consists of a disjunction of all $\phi_{a_j}(\vec{Y})$ such that the i-th bit in the binary representation of j is one. ❑

Corollary 5.5. *Let $\mathcal{C}$ be a class of languages.*

$$\text{Leaf}^P(\mathcal{C}) \subseteq Q^1_{\mathcal{C}} \Sigma^1_1$$

The second-order Lindström quantifier in the above proposition is applied to Σ^1_1-formulae in order to rule out those assignments of $\vec{Y}$ that do *not* correspond to computation paths of M. For language classes $\mathcal{C}$ that are closed under padding (see Sect. 4) we can get rid of the second-order existential quantifier. We map those assignments of $\vec{Y}$ that do not correctly represent an computation path of M to the padding symbol.

Lemma 5.6. *Let $\mathcal{C}$ be a class of languages that is closed under padding. Then*

$$\text{Leaf}^P(\mathcal{C}) = Q^1_{\mathcal{C}} \Sigma^1_0.$$

Proof. ($\subseteq$): Let M be a nondeterministic Turing machine, $B \in \mathcal{C}$, and $\phi_1(\vec{Y}), \ldots, \phi_s(\vec{Y})$ be formulae as described in Proposition 5.4 such that $\text{Leaf}^M(B) = \text{Mod}\left(Q_B X\left[\exists \vec{Y} \phi_1(\vec{Y}), \ldots, \exists \vec{Y} \phi_s(\vec{Y})\right]\right.$. We modify this formula such that the Lindström quantifier bounds $\vec{Y}$, also. As mentioned above, we map those assignments of $\vec{Y}$ that do not correctly represent an computation path of M to the padding symbol. According to Sect. 4, we have to extend the sequence $\phi_1(\vec{Y}), \ldots, \phi_s(\vec{Y})$ by an additional formula $\phi_p(\vec{Y})$. Then for all assignments $\vec{Y}^A$ the corresponding symbol in $t_\Gamma(\chi_{\phi_p^{2\mathcal{A}}}, \chi_{\phi_1^{2\mathcal{A}}}, \ldots, \chi_{\phi_s^{2\mathcal{A}}})$ is a padding symbol, iff $\phi_p(\vec{Y})$ is true for the assignment $\vec{Y}^A$. The formula $\neg\, \phi_{\text{Path}}(X, \vec{Y})$ is true, iff for given assignments $X^{\mathcal{A}}, \vec{Y}^{\mathcal{A}}$ the assignment $\vec{Y}^{\mathcal{A}}$ does not correctly represent a computation path of M with respect to the nondeterministic choices represented by $X^{\mathcal{A}}$. We therefore use $\neg\,\phi_{\text{Path}}(X,\vec{Y})$ to indicate the padding symbol. Thus we have

$$\text{Leaf}^{M}(B) = \text{Mod}\big(Q_{\text{pad}(B)}X,\vec{Y}\big[\neg\,\phi_{\text{Path}}(X,\vec{Y}),\phi_1(X,\vec{Y}),\ldots,\phi_s(X,\vec{Y})\big].$$

($\supseteq$): For proving this inclusion we do not need the fact that $\mathcal{C}$ is closed under padding. For given $B\in\mathcal{C}$ and $Q_B\vec{X}[\phi_1(\vec{X},\ldots,\phi_s(\vec{X})]$ we can construct a polynomial-time nondeterministic Turing machine M which nondeterministically guesses the assignments of $\vec{X}$ and afterwards evaluates the formulae $\phi_1(\vec{X},\ldots,\phi_s(\vec{X})$ deterministically. The resulting bit string is used to determine the correct leaf symbol according to t_Γ. ❑

Combining our leaf language characterizations of the classes of the polynomial time hierarchy (Theorem 5.3) with the just given lemma, we next examine leaf languages defined by first-order formulae starting with a Lindström quantifier.

Lemma 5.7. *Let* B *be a language. Then we have for all* $k\geq 1$*:*

$$\text{Leaf}^{P}(Q^0_B\Sigma^0_k)\subseteq Q^1_{\text{pad}(B)}\Sigma^1_k$$

Proof. We combine Lemma 5.1 and Lemma 5.6 to prove this lemma.

Let $B\in\mathcal{C}$ and $Q_B\vec{x}[\phi_1(\vec{x}),\ldots,\phi_s(\vec{x})]$ be a $Q^0_B\Sigma^0_k$-formula. As in the proof of Theorem 5.1 we may think of universes having computation paths of M as elements. We transform the first-order formulae into second-order formulae as in the proof of Theorem 5.1. In addition we transform the free variables $\vec{x}=x_1,\ldots,x_\ell$ into second-order variables $\vec{Z}=X_1,\vec{Y}_1,\ldots,X_\ell,\vec{Y}_\ell$. But as in the proof of Lemma 5.6, we have to consider those assignments for any $X_i,\vec{Y}_i$ that do not represent a computation path of M. To map those assignments to padding symbols we add a formula $\phi_p(\vec{Z})\equiv\bigvee_{1\leq i\leq\ell}\neg\phi_{\text{Path}}(X_i,\vec{Y}_i)$. We get:

$$\text{Leaf}^{M}(B) = \text{Mod}\big(Q_{\text{pad}(B)}\vec{Z}\big[\phi_p(\vec{Z}),\phi_1(\vec{Z}),\ldots,\phi_s(\vec{Z})\big].$$

❑

Lemma 5.8. *Let* B *be a language. Then we have for all* $k\geq 1$*:*

$$Q^1_B\Sigma^1_k\subseteq \text{Leaf}^{P}(Q^0_{\text{pad}(B)}\Sigma^0_k)$$

Proof. To prove this lemma we combine Lemma 5.2 and Lemma 5.6.

For given $B\in\mathcal{C}$ and formula $Q_B\vec{X}([\phi_{SO,1}(\vec{X}),\ldots,\phi_{SO,s}(\vec{X})])\in Q^1_{\mathcal{C}}\Sigma^1_k$ we construct a polynomial-time Turing machine M and a $Q^0_{\mathcal{C}}\Sigma^0_k$-formula $Q_{\text{pad}(B)}\vec{x}[\phi_p(\vec{x}),\phi_{FO,1}(\vec{x}),\ldots,\phi_{FO,s}(\vec{x})]$ such that

$$\mathrm{Mod}\big(Q_B\vec{X}[\phi_{SO,1}(\vec{X}),\dots,\phi_{SO,s}(\vec{X})]\big) \;=\; \mathrm{Leaf}^M\big(Q_{\mathrm{pad}(B)}\vec{x}[\phi_p(\vec{x}),\phi_{FO,1}(\vec{x}),\dots,\phi_{FO,s}(\vec{x})]\big).$$

Similar to the proof of Lemma 5.6 the Turing machine M guesses nondeterministically all assignments of $\vec{X}$. There is a substring of the resulting leaf string for each assignment of $\vec{X}$. These substrings start (on the left) with $\ell_{\vec{X}}$ and end (on the right) with $r_{\vec{X}}$. The first-order Lindström quantifier bounds two variables x_{left} and x_{right}. The assignments should point to the left and right boundary of some substring. We map assignments that do not correctly represent such boundaries to padding symbols as in the proof of Lemma 5.6. Let $\varphi_c(x)$ be a formula that is true iff the symbol at position x in the leaf string is c. We then construct the formula $\phi_p(x_{\mathrm{left}}, x_{\mathrm{right}})$ that indicates the padding symbol as follows:

$$\phi_p(x_{\mathrm{left}}, x_{\mathrm{right}}) \;\equiv\; \exists y(x_{\mathrm{left}} < y < x_{\mathrm{right}}) \wedge (\varphi_{\ell_{\vec{X}}}(y) \vee \varphi_{r_{\vec{X}}}(y))$$

We construct for each formula $\phi_{SO,i}(\vec{X})$ a Turing machine M_i as in the proof of Lemma 5.1. M simulates machines $M_1,\dots,M_s$ for each assignment of $\vec{X}$ as shown in Fig. 2.

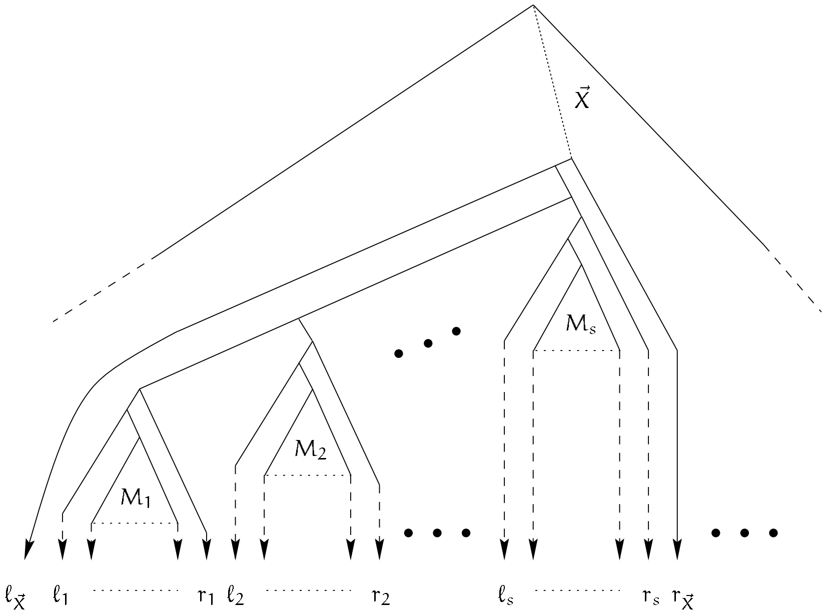

Figure 2: Construction of machine M

We now construct for each formula $\phi_{SO,i}(\vec{X})$ a first order formula $\phi_{FO,i}(\vec{x})$. As in the proof of Lemma 5.1 we construct inductively a sequence of formulae $\phi_{k,i},\dots,\phi_{2,i},\phi_{FO,i}$. The construction of $\phi_{k,i},\dots,\phi_{2,i}$ is identical to the construction given in the proof of 5.1. Let $S_{j,i}$ be the rightmost variable in the first quantifier block of $\phi_{SO,i}$ and $\mathrm{Region}_{j,i}(x, y)$ be the formula given in the proof of Theorem 5.1 that is true, iff the assignment of x and y point to the left and right boundary of the substring that corresponds to some assignment of $S_{j,i}$. By the definition of $Q^1_B \Sigma^1_k$ we may assume w. l. o. g. that $S_{j,i}$ is existentially quantified. Let $\phi_{FO,i}(x_{\mathrm{left}}, x_{\mathrm{right}})$ be the formula

$$\begin{aligned}\exists x_{\ell_i} \exists x_{r_i} \exists x_{\mathrm{left},1} \exists x_{\mathrm{right},1} \quad & (x_{\mathrm{left}} < x_{\ell_i} < x_{\mathrm{left},1} < x_{\mathrm{right},1} < x_{r_i} < x_{\mathrm{right}}) \wedge \\ & \varphi_{\ell_i}(x_{\ell_i}) \wedge \varphi_{r_i}(x_{r_i}) \wedge \\ & \mathrm{Region}_{j,i}(x_{\mathrm{left},1}, x_{\mathrm{right},1}) \wedge \phi_{2,i}(x_{\mathrm{left},1}, x_{\mathrm{right},1})\end{aligned}$$

❑

The main theorem now follows from Lemma 5.7 and Lemma 5.8.

Theorem 5.9. *Let $\mathcal{C}$ be a class of languages that is closed under padding. Then we have for all $k \geq 1$:*

$$\mathrm{Leaf}^P(Q^0_{\mathcal{C}} \Sigma^0_k) = Q^1_{\mathcal{C}} \Sigma^1_k$$

We cannot get a similar result for $Q^0_{\mathcal{C}} \Pi^1_1$-formulae since the formula $\phi_p(x_{\mathrm{left}}, x_{\mathrm{right}})$ used in the proof of Lemma 5.8 is existentially quantified. However the following certainly holds:

Corollary 5.10. *Let $\mathcal{C}$ be a class of languages that is closed under padding. Then we have for all $k \geq 2$:*

$$\mathrm{Leaf}^P(Q^0_{\mathcal{C}} \Pi^0_k) = Q^1_{\mathcal{C}} \Pi^1_k$$

Finally, we want to address an interesting special case: the class PSPACE. Barrington, Immerman, and Straubing showed that first-order logic with group quantifiers defines exactly the regular languages [BIS90]. Hertrampf et al. [HLS+93] who characterized PSPACE by regular leaf languages showed that in fact for this characterization already one single regular language, the word problem for the group S_5, is sufficient. Thus, this leaf language characterization yields the following:

Corollary 5.11. $\mathrm{PSPACE} = Q^1_{S_5} \Sigma^1_0$

6 Discussion

As we have seen, Lindström quantifiers which are a well studied logical concept have a complexity theoretic counterpart: the so called leaf language definability, which has been studied intensively in the recent past.

Second-order Lindström quantifiers define (in a model theoretic sense) exactly those languages characterizable by leaf languages for polynomial time machines. If $Q^1_{\mathcal{C}}$ is a Lindström quantifier, then the logic $Q^1_{\mathcal{C}}\Sigma^1_0$ defines the complexity class $\mathrm{Leaf}^{\mathrm{P}}(\mathcal{C})$. Thus it may be possible that results about leaf languages contribute to the study of the expressive power of second-order Lindström quantifiers on ordered finite structures, and vice versa.

Of course, it would be nice to have a leaf language analogue for first-order Lindström quantifiers. To be able to do this one will have to consider "leaf languages for FO" instead of leaf languages for polynomial time. To be more precise, what is an appropriate restriction of the computation model producing leaf words?

Gottlob in [Got95] showed that under some particular assumptions to $\mathcal{C}$, first-order formulae with arbitrarily nested $Q^0_{\mathcal{C}}$ and existential and universal quantifiers yield super-classes of L (the logspace decidable sets). Certainly, formulae with nested quantifiers should be examined along the lines of this paper, i.e. when used to define leaf languages.

Furthermore, one should consider leaf languages defined by second-order formulae to investigate the structure of complexity classes above exponential time, e.g. the exponential time hierarchy.

Acknowledgment. We thank David Barrington for suggesting the examination of logically defined leaf languages as an interesting topic. Thanks are also due to the anonymous referees for helpful comments.

References

[All96] E. ALLENDER. A note on uniform circuit lower bounds for the counting hierarchy. In *Proceedings 2nd International Computing and Combinatorics Conference (COCOON)*, volume 1090 of *Springer Lecture Notes in Computer Science*, pages 127–135, 1996.

[BCS92] D. P. BOVET, P. CRESCENZI, AND R. SILVESTRI. A uniform approach to define complexity classes. *Theoretical Computer Science*, 104:263–283, 1992.

[BIS90] D. A. M. BARRINGTON, N. IMMERMAN, AND H. STRAUBING. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41:274–306, 1990.

[BL96] B. BORCHERT AND A. LOZANO. Succinct circuit representations and leaf language classes are basically the same concept. *Information Processing Letters*, 58:211–215, 2996.

[Bur96] H. J. BURTSCHICK. *Berechnungs- und Beschreibungskomplexität von Zählfunktionen und Lindströmquantoren*. PhD thesis, Fachbereich Informatik, TU-Berlin, 1996.

[CMTV96] H. CAUSSINUS, P. MCKENZIE, D. THÉRIEN, AND H. VOLLMER. Nondeterministic NC^1 computation. In *Proceedings 11th Computational Complexity*, pages 12–21, 1996. To appear in *Journal of Computer and System Sciences.*

[Ebb85] H.-D. EBBINGHAUS. Extended logics: The general framework. In J. Barwise and S. Feferman, editors, *Model-Theoretic Logics*, Perspectives in Mathematical Logic, chapter II, pages 25–76. Springer Verlag, 1985.

[EF95] H. D. EBBINGHAUS AND J. FLUM. *Finite Model Theory*. Springer, 1995.

[Fag74] R. FAGIN. Generalized first-order spectra and polynomial time recognizable sets. In R. Karp, editor, *Complexity of Computations*, pages 43–73, 1974.

[Got95] G. GOTTLOB. Relativized logspace and generalized quantifiers over finite structures. Technical Report CD-TR-95/76, Institut for Information Systems, Vienna University of Technology, 1995. An extended abstract appeared in the proceedings of the 10th Symposium on Logic in Computer Science, 1995.

[Her95] U. HERTRAMPF. Regular leaf-languages and (non-) regular tree shapes. Technical Report A-95-21, Institut für Mathematik und Informatik, Medizinische Universität zu Lübeck, 1995.

[HLS+93] U. HERTRAMPF, C. LAUTEMANN, T. SCHWENTICK, H. VOLLMER, AND K. W. WAGNER. On the power of polynomial time bit-reductions. In *Proceedings 8th Structure in Complexity Theory*, pages 200–207, 1993.

[HVW95] U. HERTRAMPF, H. VOLLMER, AND K. W. WAGNER. On the power of number-theoretic operations with respect to counting. In *Proceedings 10th Structure in Complexity Theory*, pages 299–314, 1995.

[HVW96] U. HERTRAMPF, H. VOLLMER, AND K. W. WAGNER. On balanced vs. unbalanced computation trees. *Mathematical Systems Theory*, 29:411–421, 1996.

[JMT94] B. JENNER, P. MCKENZIE, AND D. THÉRIEN. Logspace and logtime leaf languages. In *Proceedings 9th Structure in Complexity Theory*, pages 242–254, 1994.

[Lin66] P. LINDSTRÖM. First order predicate logic with generalized quantifiers. *Theoria*, 32:186–195, 1966.

[Pap94] C. H. PAPADIMITRIOU. *Computational Complexity*. Addison-Wesley, 1994.

[Sto73] L. STOCKMEYER. The polynomial-time hierarchy. *Theoretical Computer Science*, 3:1–22, 1973.

[Str94] H. STRAUBING. *Finite Automata, Formal Logic, and Circuit Complexity*. Birkhäuser, 1994.

[Vei96] H. VEITH. Succinct representation, leaf languages, and projection reductions. In *Proceedings 10th Structure in Complexity Theory*, pages 118–126, 1996.

[Ver93] N. K. VERESHCHAGIN. Relativizable and non-relativizable theorems in the polynomial theory of algorithms. *Izvestija Rossijskoj Akademii Nauk*, 57:51–90, 1993. In Russian.