Succinct Inputs, Lindström Quantifiers, and a General Complexity Theoretic Operator Concept

Heribert Vollmer

Theoretische Informatik
Universität Würzburg
Am Exerzierplatz 3
D-97072 Würzburg, Germany

vollmer@informatik.uni-wuerzburg.de

Abstract

We address the question of the power of several logics with Lindström quantifiers over finite ordered structures.

We will see that in the first-order case this nicely fits into the framework of Barrington, Immerman, and Straubing's examination of constant depth circuit classes.

In the second-order case we get a strong relationship to succinct encodings of languages via circuits. Some of these logics can be characterized as closures of succinct encodings under appropriate reducibilities, others by certain hierarchies of circuit classes. We will see that in a special case second-order Lindström quantifiers can equivalently be expressed in first-order logic, while in the general case this equivalence seems unlikely.

1 Introduction

The question attacked in this paper is: What is the expressive power of logics with generalized quantifiers over finite ordered structures?

The formal framework for general quantifiers we use is the one introduced by Lindström in 1966 (Lindström, 1966). He showed how every problem (i.e., every class of structures over some fixed signature) gives rise to the definition of a quantifier. For Q such a quantifier defined by the structure class $\mathcal{K}$ we denote by Q^+ the set of all formulae $Q\vec{x}\phi(\vec{x})$, where ϕ is first-order. Now it can be seen that Q^+ essentially characterizes all problems reducible to $\mathcal{K}$ via first-order reductions as defined e.g. in (Ebbinghaus and Flum, 1995).

A lot of special cases have been examined in the literature. Immerman's transitive closure operator TC (Immerman, 1987) is nothing else than the Lindström quantifier obtained from the transitive closure problem. (Observe that transitive closure is not expressible in first-order logic.) Henkin quantifiers H have been studied intensively, e.g. in (Blass and Gurevich, 1986). Stewart examined several quantifiers defined by NP-complete problems, see for instance (Stewart, 1991; Stewart, 1992).

What is the expressive power of these quantifiers over finited ordered structures? Say that a logic $\mathcal{L}$ captures a complexity class $\mathcal{K}$, if $\mathcal{L}$ formulae can be evaluated in $\mathcal{K}$ and every $\mathcal{K}$ decidable property can be expressed by some $\mathcal{L}$ sentence. It is known that TC^+ captures nondeterministig logspace, and H^+ as well as the quantifiers considered by Stewart capture NP.

Gottlob in 1995 addressed the following question: Given that Q^+ captures $\mathcal{K}$, what class does $FO(Q)$ (i.e., full first-order logic with possibly nested applications of Q quantifiers) capture? What he proved is that under a technical assumption, the so called *smoothness* of $\mathcal{K}$, if Q^+ captures $\mathcal{K}$, then $FO(Q)$ captures $L^{\mathcal{K}}$.

Smoothness however is a property designed for large classes; "large" here means: logspace or superclasses of logspace. But there are a lot of quantifiers Q defined by regular languages for which we are in a lower range, usually within NC^1. Immerman proved that first-order logic captures AC^0, but for this result we have to include the bit predicate in our logic (more on this below). Barrington and Therien (Barrington and Thérien, 1988) examined subclasses of NC^1 defined via monoid programs. As it turned out, these classes can be characterized by first-order logic equipped with monoidal quantifiers (Barrington et al., 1990). Thus we see that there is a relation between subclasses of NC^1 and first-order logic with generalized quantifiers. It can be observed that this latter result does not require that the quantifier is given by a monoid word problem. One can show that for a very general class of quantifiers, $FO(Q)$ coincides with (logtime-uniform) AC^0 circuits with Q gates in addition to AND and OR gates. However, for all these results we have to include in our logic the bit predicate (see the discussion in (Immerman, 1989) for an intuitive explanation; see also (Lindell, 1992)). Since we are interested in small circuit classes in this paper, we follow this assumption.

Burtschick and Vollmer defined and examined second-order Lindström quantifiers (Burtschick and Vollmer, 1996). While first-order quantifiers are defined by classes of structures (i.e., sequences of relations), these second-order quantifiers are defined by second-order structures, i.e. classes of (sequences of) sets of relations. It was shown that with these quantifiers, the concept of leaf-language definability can be characterized in a purely logical way. Given a set B, the logic consisting of all formulae starting with a second-order quantifier given by B followed by a formula without other second-order quantifiers captures exactly the complexity class defined by the leaf language B on polynomial time machines, i.e. $\mathrm{Leaf}^{\mathrm{P}}(B)$.

In this paper we examine some of the cases which are not covered by the above cited papers. In the case of second-order Lindström quantifiers, we examine the power of full second-order logic with nested applications of the quantifier, as well as full first-order logic with nested applications of the second-order quantifier. In the case of first-order Lindström quantifiers, we relate the class Q^+ with leaf languages for logtime machines and show how this allows us to obtain characterizations for some of these logics.

Veith (Veith, 1996) and Borchert and Lozano (Borchert and Lozano, 1996) related leaf language defined classes with succinct input representations. (Borchert and Lozano, 1996) showed that $\mathrm{Leaf}^{\mathrm{P}}(B)$ is exactly the closure of the succinct version of B under polynomial time many-one reducibility. Veith obtained that this holds even for first-order projection reducibility.

Here we relate these results with the above mentioned logical characterization of the leaf language concept by Burtschick and Vollmer. This will allow us to characterize some logics with second-order Lindström quantifiers as redubility closure of succint problem encodings, and moreover to express these logics using only first-order quantifiers.

To formulate some of our results we introduce a general complexity theoretic operator concept. This concept will allow us to talk easily about hierarchies of classes defined by some accepting condition on, say, nondeterministic polynomial time machines.

There are several other approaches to the examination of the expressive power of generalized quantifiers. A lot of pointers to the literature can be found in (Gottlob, 1995b). Our paper follows mainly Gottlob's approach, but we include in our logic the bit predicate as in (Immerman, 1989; Barrington et al., 1990; Barrington and Immerman, 1994).

2 Preliminaries

Signatures and Structures

A signature is a finite sequence $\tau = \langle R_1, \dots, R_k, c_1 \dots, c_\ell \rangle$ of relation symbols and constant symbols. A finite structure of signature τ is a tuple $\mathcal{A} = (A, R_1^{\mathcal{A}}, \dots, R_k^{\mathcal{A}}, c_1^{\mathcal{A}} \dots, c_\ell^{\mathcal{A}})$ consisting of a finite set A (the universe of $\mathcal{A}$) and interpretations of the symbols in τ by relations over $\mathcal{A}$ (of appropriate arity) and elements of $\mathcal{A}$. In this paper we only consider ordered structures. We may assume w.l.o.g. that the universe of every structure $\mathcal{A}$ is always an initial set of the natural numbers. $\mathrm{Struct}(\tau)$ is the set of all *finite ordered structures* over τ.

Let e denote one of the standard encodings of structures into binary strings, e.g. fix $e(\mathcal{A})$ to be $|A|$ in unary followed by the characteristic strings of the relations in $\mathcal{A}$ followed by the binary encodings of the constant elements. Here, the *characteristic string* χ_R of a relation $R \in \{0, \dots, n-1\}^a$ is the string $\chi_R =_{\mathrm{def}} b_1 \cdots b_{n^a}$ where $b_i = 1$ iff the i-th vector in

$\{0,\dots,n-1\}^a$ (in the order $(0,\dots,0,0) < (0,\dots,0,1) < (n-1,\dots,n-1,n-1)$) is in R. For $1 \le i \le n$, let $\chi_R[i]$ denote the i-th bit in χ_R.

FO (SO) denotes the set of all first-order (second-order) sentences over τ with predicates for equality, a less-than relation, and the bit predicate. (We will say more about the bit predicate in Sect. 4.)

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, and on the other hand for every fixed $\mathcal{L}$ sentence ϕ, determining whether $\mathcal{A} \models \phi$ can be done in $\mathcal{K}$. As an abbreviation we will most of the time simply write $\mathcal{K} = \mathcal{L}$.

Generalized 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 (Ebbinghaus and Flum, 1995).

Every class of structures $K \subseteq \text{Struct}(\sigma)$ over a signature $\sigma = \langle P_1,\dots,P_s\rangle$ defines the first-order Lindström quantifier Q_K as follows: Let $\phi_1,\dots,\phi_s$ be first-order formulae over signature τ 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 $Q_K\vec{x}_1,\dots,\vec{x}_s\,[\phi_1(\vec{x}_1),\dots,\phi_s(\vec{x}_s)]$ is a Q_KFO formula. If $\mathcal{A} \in \text{Struct}(\tau)$, then $\mathcal{A} \models Q_K\vec{x}_1,\dots,\vec{x}_s\,[\phi_1(\vec{x}_1),\dots,\phi_s(\vec{x}_s)]$ iff $(A,\phi_1^{\mathcal{A}},\dots,\phi_s^{\mathcal{A}}) \in K$.

The just given definition is the original definition given by Lindström (Lindström, 1966), which the reader will also find in textbooks (Ebbinghaus, 1985; Ebbinghaus and Flum, 1995) and formulated with slight variations but equivalently in (Gottlob, 1995b). 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 ($n = |A|$). Now given a sequence $\phi_1,\dots,\phi_s$ of formulae with k free variables each and a structure $\mathcal{A}$, we similarly get the tuple $(\chi_{\phi_1^{\mathcal{A}}},\dots,\chi_{\phi_s^{\mathcal{A}}})$, 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. $A_s = \{a_0,a_1,\dots,a_{2^s-1}\}$. Then $(\chi_{\phi_1^{\mathcal{A}}},\dots,\chi_{\phi_s^{\mathcal{A}}})$ corresponds to the string $b_1b_2\cdots b_{n^k}$, 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: $w_s(\chi_{\phi_1^{\mathcal{A}}},\dots,\chi_{\phi_s^{\mathcal{A}}}) = b_1b_2\cdots b_{n^k}$.

This leads us to the following definition: A sequence $[\phi_1,\dots,\phi_s]$ is in *first-order word normal form,* iff the ϕ_i have the same number k of free variables. Let Γ be an alphabet such that $|\Gamma| \le 2^s$, and let $B \subseteq \Gamma^*$. Then $\mathcal{A} \models Q_B\vec{x}\,[\phi_1(\vec{x}),\dots,\phi_s(\vec{x})]$ iff $w_s(\chi_{\phi_1^{\mathcal{A}}},\dots,\chi_{\phi_s^{\mathcal{A}}}) \in B$.

It can be shown (Burtschick, 1996) that every Lindström quantifier Q_K can without loss of generality be assumed to be of the form Q_B as just defined. This is the case since for every sequence $[\phi_1,\dots,\phi_s]$ of first-order formulae we find an equivalent sequence in word normal form such that the corresponding formulae with Lindström quantifier express the same property.

Second-order Lindström quantifiers are defined as follows (as already sketched in (Burtschick and Vollmer, 1996), a sound definition appears in (Burtschick, 1996)): Given a formula ϕ with free second-order variables $P_1, \dots, P_m$ and a structure $\mathcal{A}$, define $\phi^{2^{\mathcal{A}}} =_{\text{def}} \{ (R_1^{\mathcal{A}}, \dots, R_m^{\mathcal{A}}) \mid \mathcal{A} \models \phi(R_1^{\mathcal{A}}, \dots, R_m^{\mathcal{A}}) \}$, 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 $P_1, \dots, P_m$ are $r_1, \dots, r_m$, resp., then the length of $\chi_{\phi^{2^{\mathcal{A}}}}$ is $2^{n^{r_1}+\dots+n^{r_m}}$

Let $\sigma = \langle \sigma_1, \dots, \sigma_s \rangle$ be a signature, where $\sigma_i = \langle P_{i,1}, \dots, P_{i,m_i} \rangle$ for $1 \le i \le s$. Thus σ is a signature consisting of 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 $\mathcal{A} = (A, \mathcal{R}_1, \dots, \mathcal{R}_s)$, where for every $1 \le i \le s$, $\mathcal{R}_i \subseteq \{ (R_{i,1}, \dots, R_{i,m_i}) \mid R_{i,j} \subseteq A^{\ell_{i,j}} \}$. Given now a signature τ and second-order formulae $\phi_1(\vec{X}_1), \dots, \phi_s(\vec{X}_s)$ over τ where for every $1 \le i \le s$ 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 $Q_{\mathcal{K}}\vec{X}_1, \dots, \vec{X}_s \left[\phi_1(\vec{X}_1), \dots, \phi_s(\vec{X}_s)\right]$ is a $Q_{\mathcal{K}}$SO formula. If $\mathcal{A} \in \text{Struct}(\tau)$, then $\mathcal{A} \models Q_{\mathcal{K}}\vec{X}_1, \dots, \vec{X}_s \left[\phi_1(\vec{X}_1), \dots, \phi_s(\vec{X}_s)\right]$ iff $(A, \phi_1^{2^{\mathcal{A}}}, \dots, \phi_s^{2^{\mathcal{A}}}) \in \mathcal{K}$.

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 $\sigma_1 = \dots = \sigma_s = \langle P_1, \dots, P_m \rangle$. Let for $1 \le i \le m$ the arity of P_i be r_i. Observe that in this case, $|\chi_{\phi_1^{2^{\mathcal{A}}}}| = \dots = |\chi_{\phi_s^{2^{\mathcal{A}}}}| = 2^{n^{r_1}+\dots+n^{r_m}}$ (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. Given now a language $B \subseteq \Gamma^*$ with $|\Gamma| \le 2^s$, the second-order Lindström quantifier given by B is defined by $\mathcal{A} \models Q_B^1\vec{X} \left[\phi_1(\vec{X}_1, \dots, \phi_s(\vec{X}_s)\right]$ iff $w_s(\chi_{\phi_1^{2^{\mathcal{A}}}}, \dots, \chi_{\phi_s^{2^{\mathcal{A}}}}) \in B$.

Again it was shown in (Burtschick, 1996) that for every second-order Lindström quantifier $Q_{\mathcal{K}}$ there is an equivalent Q_B^1.

When talking about the first-order Lindström quantifier given by B, we sometimes explicitly write Q_B^0 instead of Q_B.

In addition to the above logics Q_B^0FO and Q_B^1SO where we allow Lindström quantifiers followed by an arbitrary first-order (second-order, resp.) formula, we also need Q_B^1FO (where we have a second-order Lindström quantifier followed by a formula with no other second-order quantifiers), and FO(Q_B^0) and SO(Q_B^1) (where we have first-order (second-order, resp.) formulae with arbitrary nesting of universal, existential, and Lindström quantifiers). Following (Gottlob, 1995b), we also write Q_B^+ for Q_B^0FO. For a class of languages $\mathcal{C}$ we use the notation $Q_{\mathcal{C}}$ with the obvious meaning, e.g. FO$(Q_{\mathcal{C}}^0)$ denotes all first-order sentences with arbitrary quantifiers Q_B^0 for $B \in \mathcal{C}$.

Succinct Representations

A succint representation (Wagner, 1986; Balcázar et al., 1992; Veith, 1996) of a binary word x is a boolean circuit given on input i the ith bit of x. The succinct version sA of a language A is the following: Given a boolean circuit describing a word x, is $x = x_1 0 x_2 0 \cdots x_{n-1} 0 x_n 1 w$ for arbitrary $w \in \{0,1\}^*$, such that $x_1 x_2 \cdots x_n \in A$? The boolean circuits we allow are standard unbounded fan-in circuits over AND, OR, NOT. The encoding consists of a sequence of tuples (g, t, h), where g and h are gates, t is the type of g, and h is an input gate to g (if g is not already an input variable).

Let $\mathcal{N}$ be the class of all languages having a *neutral element*, i.e. if $A \subseteq \Sigma^*$, then $A \in \mathcal{N}$ if there is a letter $e \in \Sigma$ sucht that for every $u, v \in \Sigma^*$, $uev \in A \iff uv \in A$.

When encoding circuits we will presuppose some alphabet for the encoding with a neutral element. This assumption is similar in effect to the use of so called implicit circuits in (Veith, 1996). It is obvious that a succinct representation of a word under this assumption (with neutral element) first-order reduces to the succinct representation as before. Thus it seems fairly natural and intuitive to make this assumption, and we will use again the notation sA for this encoding. Thus in the following, for every set A, we have $sA \in \mathcal{N}$. It should be remarked however that all our results below (except possibly Theorem 5.3) hold also for the above original encoding without neutral element.

Leaf Languages

The definability of complexity classes via leaf languages was introduced in (Bovet et al., 1992; Vereshchagin, 1993) (see also the recent textbook (Papadimitriou, 1994)). 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 $\text{Leaf}^M(B) =_{\text{def}} \{\, x \mid \beta_M(x) \in B \,\}$. The class $\text{Leaf}^P(B)$ is the class of all languages $\text{Leaf}^M(B)$ for all nondeterministic polynomial-time machines M. Here, B is the so called *leaf language* defining the class $\text{Leaf}^P(B)$. If $\mathcal{C}$ is a class of languages, then we define $\text{Leaf}^P(\mathcal{C}) =_{\text{def}} \bigcup_{B \in \mathcal{C}} \text{Leaf}^P(B)$.

We will in the following mainly consider leaf languages $B \in \mathcal{N}$. As it turned out (Jenner et al., 1994) in this case the above definition is equivalent to the more restricted one where we only consider Turing machines with full binary trees.

We also use leaf languages for log time machines. Our notation there is $\text{Leaf}^{LT}(B)$. We use Turing machines with a random access input tape as model for log time computation. The machines may query the input at any time and are charged one time unit for each query. The index tape is not erased after a query. For background on this model as well as a comparison with other possible definitions of logtime see (Regan and Vollmer, 1996).

3 Polynomial Time vs. Second-Order

In (Burtschick and Vollmer, 1996; Burtschick, 1996) the concept of leaf language definability was characterized in a logical way. It was shown that for $B \in \mathcal{N}$, $\text{Leaf}^P(B) = Q^1_B\text{FO}$.

If the leaf language is itself given in a logical way, then the defined class is logically characterizable by replacing first-order by second-order quantifiers, thus for $B \in \mathcal{N}$, we have $\mathrm{Leaf}^{P}(Q_B^+) = Q_B^1 \mathrm{SO}$. It can be seen (see e.g. (Immernan and Landau, 1995; Gottlob, 1995b)) that $Q_B^+ = \mathcal{K}$ iff B is complete for $\mathcal{K}$ under first-order reductions. Analogously we obtain:

Theorem 3.1 *If* $B \in \mathcal{N}$, *then* $Q_B^1\mathrm{FO} = \mathcal{K}$ *iff* sB *is complete for* $\mathcal{K}$ *under first-order reductions.*

Proof sketch. Follows from the above cited results from (Burtschick and Vollmer, 1996) and Theorem 3 from (Veith, 1996). ❑

As an immediate consequence of Theorem 3.1 we get a characterization of second-order Lindström quantifiers by first-order logics.

Corollary 3.2 *If* $B \in \mathcal{N}$, *then* $Q_B^1\mathrm{FO} = Q_{sB}^+$.

Thus we get the following special cases from well-known leaf language characterizations of PSPACE (Hertrampf et al., 1993):

Corollary 3.3 *The succinct version of the accessibility problem for directed graphs of bounded width as well as the succinct version of the word problem of any nonsolvable monoid are* PSPACE*-complete under* FO *reductions.*

In a similar spirit we can use other characterizations, e.g. those in (Hertrampf, 1996), to show completeness of succinct word problems of certain monoids for classes like Δ_2^p, $\oplus\mathrm{P}$, $\mathrm{P}^{\mathrm{NP}[1]:\oplus\mathrm{P}[1]}$, etc. On the other hand, we can use Hertrampf's result to obtain logical characterizations of these classes, e.g. $\mathrm{P}^{\mathrm{NP}[1]:\oplus\mathrm{P}[1]}$ can be characterized by a certain first-order Lindström quantifier.

Similarly we can use completeness results for certain succinct problems, e.g. those in (Wagner, 1986; Balcázar et al., 1992), to obtain immediately logical characterizations of the corresponding classes, either with first-order or with second-order Lindström quantifier.

A naturally arising question now is of course: what about nested quantifiers? We come back to this issue later.

4 Constant Depth vs. First-Order

Barrington, Immerman, and Straubing in (Barrington et al., 1990) (see also (Barrington and Immerman, 1994)) showed that if B is a word problem of a monoid M, then $\mathrm{FO}(Q_B^0) = \mathrm{AC}^0[B]$, i.e. AC^0 circuits with gates for multiplication in M. As can be seen from their proof, the result holds generally for any $B \in \mathcal{N}$ (see also (Bédard et al., 1993)). This equality, as well as all our results below, hold for DLOGTIME- as well as FO-uniformity. Since multiplication in B is not necessarily commutative we have to specify in our circuit encodings an order of the inputs of B gates. This leads to a natural generalization of the "direct connection language", see (Ruzzo, 1981; Barrington et al., 1990; Barrington and Immerman, 1994). However, the result differs from those in the previous section since here we *have* to include in our language the bit predicate (see the discussion in (Immerman, 1989)). We remark that the results in the previous section hold regardless of the presence of a bit predicate.

It is known that in a number of special cases, if Q_B^+ captures $\mathcal{K}$, then $\mathrm{FO}(Q_B^0)$ captures $L^{\mathcal{K}}$. Gottlob identified a necessary and sufficient condition for this to happen, the so called *smoothness* of $\mathcal{K}$ (Gottlob, 1995b). It follows that if $\mathcal{K}$ is smooth, then $\mathrm{AC}^0[\mathcal{K}] = L^{\mathcal{K}}$.

But how to determine the power of the subclass Q_B^+? Intuitively, we would expect a correspondence to AC^0 circuits "with a B gate on top." To make a notion like this precise for general classes (not necessarily defined by circuits), we define (c_A denotes the characteristic function of set A):

Definition 4.1 Let $B \subseteq \{0,1\}^*$ be a language, $\mathcal{F}$ a class of functions on natural numbers and $\mathcal{K}$ be a class of languages. Then $A \in (B)^{\mathcal{F}}\mathcal{K}$ iff there is an $f \in \mathcal{F}$ and a set $D \in \mathcal{K}$ such that for all x we have $x \in A \iff c_D(x, 0^{f(|x|)})c_D(x, 0^{f(|x|)-1}1)\cdots c_D(x, 1^{f(|x|)}) \in B$.

If B is not a language over the binary alphabet, but $B \subseteq \{a_0, \dots, a_m\}^*$, then we use block encoding to associate with $c_D(x, 0^{f(|x|)})c_D(x, 0^{f(|x|)-1}1)\cdots c_D(x, 1^{f(|x|)})$ a word to be checked for membership in B.

Let $(\mathcal{C})^{\mathcal{F}}\mathcal{K} =_{\mathrm{def}} \bigcup_{B\in\mathcal{C}}(B)^{\mathcal{F}}\mathcal{K}$. If $\mathcal{C}$ is a finite set $\{A_1, \dots, A_k\}$, then we also use the notation $(A_1, \dots, A_k)^{\mathcal{F}}\mathcal{K}$ for $(\mathcal{C})^{\mathcal{F}}\mathcal{K}$.

We write $(B)^p\mathcal{K}$ if $\mathcal{F}$ is the class of all polynomials, and $(B)^\ell\mathcal{K}$ if $\mathcal{F}$ is the class of all functions $c \cdot \log n$ for some $c \in \mathbb{N}$.

It is obvious that for every $B \in \mathcal{N}$, $\mathrm{Leaf}^{\mathrm{P}}(B) = (B)^p\mathrm{P}$. Thus what we defined is in a sense a generalization of the leaf language concept.

A reformulation of Burtschick and Vollmer's above mentioned result now is:

Proposition 4.2 *For* $B \in \mathcal{N}$, $Q_B^1\mathrm{SO} = (B)^p\mathrm{PH}$.

Moreover, one can show additionally to their $Q_B^1\mathrm{FO} = (B)^p\mathrm{P}$:

Theorem 4.3 *For* $B \in \mathcal{N}$, $Q_B^1\mathrm{FO} = (B)^p\mathrm{AC}^0$.

Proof sketch. The left to right inclusion follows essentially from Immerman's $\mathrm{FO} = \mathrm{AC}^0$. The right hand class is certainly a subclass of $(B)^p\mathrm{P}$ which, by the above, coincides with $Q_B^1\mathrm{FO}$. ❑

This shows us that leaf language classes can not only be characterized by the above operator applied to the base class P, but already AC^0 is sufficient, i.e., for $B \in \mathcal{N}$, $\mathrm{Leaf}^{\mathrm{P}}(B) = (B)^p\mathrm{AC}^0$.

Now the following holds:

Theorem 4.4 *If* $B \in \mathcal{N}$*, then* $Q_B^+ = (A)^\ell AC^0 = (A)^\ell LH$.

Proof sketch. Follows from $FO = AC^0 = LH$ (Immerman, 1989; Sipser, 1983) and some additional considerations, especially taking care of uniformity. ❑

The previous result, though it looks rather tautological, allows us to identify in an easy way classes of the form Q_B^+ by making use of complexity theoretic results on logtime leaf languages and logtime reducibility, since we get the following lower bound for Q_B^+:

Corollary 4.5 *If* $B \in \mathcal{N}$*, then* $\mathrm{Leaf}^{LT}(B) \subseteq Q_B^+$.

Thus we obtain the following characterizations. Let Groupoid denote the class of all groupoid word problems (Bédard et al., 1993). Let G denote Greibach's hardest context-free language, see e.g. (Wagner and Wechsung, 1986).

(Bédard et al., 1993) showed that $FO(Q_{\mathrm{Groupoid}}) = \mathrm{LOGCFL}$. Our statement 2 below shows that already Q_{Groupoid}^+ is sufficient.

Corollary 4.6 *1.* $\mathrm{PSPACE} = Q_{\mathrm{CSL}}^+ = Q_{\mathrm{INEQ}}^+ = FO(Q_{\mathrm{CSL}})$.

2. $\mathrm{LOGCFL} = Q_{\mathrm{CFL},}^+ = Q_{\mathrm{Groupoid}}^+ = Q_G^+ = FO(Q_{\mathrm{Groupoid}})$.

Proof sketch. We prove the inclusions from left to right.

The first inclusion follows in both statements from characterizations given in (Jenner et al., 1994), and Corollary 4.5. The INEQ problem is complete for CSL under uniform projections (Wagner and Wechsung, 1986) which proves the second inclusion in 1. Context-free languages are reducible to groupoid word problems by homomorphisms (Bédard et al., 1993), which proves the second inclusion in 2. Greibach's language G is complete for CFL under first-order reductions (Wagner and Wechsung, 1986), which proves the third inclusion in 2. The last inclusion in both statements is obvious, as well as the inclusion of the rightmost class in the leftmost. ❑

5 Second-Order Quantifiers Revisited

We now also address the question of the power of logics with nested second-order Lindström quantifiers. Let us first assume that there are no other second-order quantifiers. We denote the corresponding logic by $FO(Q_B^1)$. We obtain:

Theorem 5.1 *For* $B \in \mathcal{N}$*,* $FO(Q_B^1)$ *is equal to the* $(B)^p AC^0$ *oracle hierarchy, i.e.* $(B)^p AC^0 \cup (B)^p AC^0[(B)^p AC^0] \cup (B)^p AC^0[(B)^p AC^0[(B)^p AC^0]] \cup \cdots$.

Proof sketch. We have to show that for any language A, the following three statements are equivalent:

(i) $A \in \mathrm{FO}(Q^1_B)$.

(ii) A is in the DLOGTIME-uniform $(B)^p\mathrm{AC}^0$ oracle hierarchy.

(iii) A is in the FO-uniform $(B)^p\mathrm{AC}^0$ oracle hierarchy.

The implication (i) $\implies$ (ii) is a simple induction, replacing the logical connectives by appropriate circuit gates, universal and existential quantifiers by unbounded fan-in AND and OR gates, and Q^1_B quantification by subcircuits with special input gates.

The implication (ii) $\implies$ (iii) follows from the fact that DLOGTIME is a subclass of FO.

The implication (iii) $\implies$ (i) is proved as follows:

As a first step we replace inductively oracle gates by plugging in the corresponding circuit as a subcircuit. This yields a polynomial size constant depth circuit with B nodes as inner gates.

We can refer to node numbers in this circuit as tuples of variables, since we deal with a constant number of polynomial size circuits. We define inductively, as in (Barrington et al., 1990), a predicate $\mathrm{Acc}_{c,d}(g)$ expressing that g is a gate on circuit level c in an oracle circuit at hierarchy level d that evaluates to 1. The cases of AND, OR, NOT, and input gates are treated as in (Barrington et al., 1990). Input gates in subcircuits are replaced by the corresponding $\mathrm{Acc}_{c,d}$ predicate. B gates can be expressed as a Q^1_B quantification over the accept predicate for the preceding level, where we let these predicates evaluate to the neutral element for B if the gate is not fed as an input to the B gate (observe that the circuit's direct connection language is in FO). Access to the special input gates (there are at most polynomially many) in the oracle subcircuits are replace by referring to a Q^1_B quantified relation of appropriate arity. ❑

Define the language E by the regular expression $(0+1)^*1(0+1)^*$. Then $\mathrm{FO}(Q^1_E) = \mathrm{SO}$ which captures PH, and $\mathrm{FO}(Q^1_{\{B,E\}}) = \mathrm{SO}(Q^1_B)$. What we get from the previous result is that this captures the oracle hierarchy having as building blocks the classes $\mathrm{Leaf}^{\mathrm{P}}(B)$ and NP. Thus, here we are again roughly in the range of polynomial time classes, but we have to be careful, since the oracle access mechanism is restricted as in Theorem 5.1. Thus, we like to express our result as follows:

Theorem 5.2 *If* $B \in \mathcal{N}$, *then* $\mathrm{SO}(Q^1_B)$ *captures the oracle hierarchy built by* $(B)^p\mathrm{AC}^0$ *and* $(E)^p\mathrm{AC}^0$.

As mentioned before, this is in general not equal to the hierarchy built by NP and $\mathrm{Leaf}^{\mathrm{P}}(B)$ with unrestricted oracle access. However, in certain special cases we get equality: Define e.g. $U =_{\mathrm{def}} (0^*10^*1)^*0^*10^*$; then $\mathrm{SO}(Q^1_U)$ is the NP-$\oplus$P oracle hierarchy. As follows from Toda's result (Toda, 1991) this hierarchy collapses to the $\exists\forall\oplus$-level; thus we get a normal form for the corresponding logic.

What remains to examine is the power of first-order with nested applications of Q^0_{sB} quantifiers.

Theorem 5.3 $\mathrm{FO}(Q^0_{sB}) = \mathrm{AC}^0[sB]$.

Proof sketch. Follows from $\mathrm{FO}(Q_A) = \mathrm{AC}^0[A]$ for $A \in \mathcal{N}$ (Barrington et al., 1990). ❑

For example in the case of the language E we have that sE is NP-complete, thus $\mathrm{FO}(Q^0_{sE})$ captures $\mathrm{AC}^0[\mathrm{NP}]$, which is known to be equal to $\Theta^p_2 = \mathrm{L}^{\mathrm{NP}}$, see (Gottlob, 1995a).

One should contrast this with Corollary 3.2, which showed that in the base step the quantifiers Q^0_{sB} and Q^1_B are equivalent in the sense that $Q^0_{sB}\mathrm{FO} = Q^1_B\mathrm{FO}$. This equivalence probably no longer holds when nesting is allowed, as the mentioned examples show.

In the second-order case we get:

Theorem 5.4 *If* $B \in \mathcal{N}$, *then* $\mathrm{SO}(Q^0_{sB}) = \mathrm{PH}^{sB}$.

6 Conclusion

We addressed the question of the relation of logics with Lindström quantifiers to complexity classes. We got the following picture (most equalities only known to hold for $B \in \mathcal{N}$):

$Q^1_B\mathrm{SO}$	$=$	$(B)^p\mathrm{PH}$	$\mathrm{SO}(Q^1_B)$	$=$	$(B,E)^p\mathrm{AC}^0$-hierarchy
$Q^1_B\mathrm{FO}$	$=$	$(B)^p\mathrm{P} = (B)^p\mathrm{AC}^0$	$\mathrm{FO}(Q^1_B)$	$=$	$(B)^p\mathrm{AC}^0$-hierarchy
	$=$	$Q^0_{sB}\mathrm{FO}$	$\mathrm{FO}(Q^0_{sB})$	$=$	$\mathrm{AC}^0[sB]$
Q^+_B	$=$	$(B)^\ell\mathrm{AC}^0 = (B)^\ell\mathrm{LH}$	$\mathrm{FO}(Q^0_B)$	$=$	$\mathrm{AC}^0[B]$

We proved that these connections between finite model theory and complexity theory can be conveniently exploited to transfer results from one side to the other by giving completeness results as well as logical characterizations for a number of prominent classes, and as briefly pointed out Sect. 5 to obtain normal forms for certain logics.

To formulate our results we made use of a general operator concept. This paper is of course only a small first step towards the examination of this notion, but we think this might turn out to be a worthwhile study.

References

Balcázar, J. L., Lozano, A., and Torán, J. (1992). The complexity of algorithmic problems on succinct instances. In Baeza-Yates, R. and Manber, U., editors, *Computer Science*. Plenum Press, New York.

Barrington, D. A. M. and Immerman, N. (1994). Time, hardware, and uniformity. In *Proceedings 9th Structure in Complexity Theory*, pages 176–185. IEEE Computer Society Press.

Barrington, D. A. M., Immerman, N., and Straubing, H. (1990). On uniformity within NC^1. *Journal of Computer and System Sciences*, 41:274–306.

Barrington, D. A. M. and Thérien, D. (1988). Finite monoids and the fine structure of NC^1. *Journal of the Association of Computing Machinery*, 35:941–952.

Bédard, F., Lemieux, F., and McKenzie, P. (1993). Extensions to Barrington's M-program model. *Theoretical Computer Science*, 107:31–61.

Blass, A. and Gurevich, Y. (1986). Henkin quantifiers and complete problems. *Annals of Pure and Applied Logic*, 32:1–16.

Borchert, B. and Lozano, A. (1996). Succinct circuit representations and leaf language classes are basically the same concept. *Information Processing Letters*, 58:211–215.

Bovet, D. P., Crescenzi, P., and Silvestri, R. (1992). A uniform approach to define complexity classes. *Theoretical Computer Science*, 104:263–283.

Burtschick, H. J. (1996). *Berechnungs- und Beschreibungskomplexität von Zählfunktionen und Lindströmquantoren*. PhD thesis, Fachbereich Informatik, TU-Berlin.

Burtschick, H.-J. and Vollmer, H. (1996). Lindström quantifiers and leaf language definability. Technical Report 96-005, Electronic Colloquium on Computational Complexity. Submitted for publication.

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

Ebbinghaus, H.-D. and Flum, J. (1995). *Finite Model Theory*. Springer.

Gottlob, G. (1995a). NP trees and Carnap's modal logic. *Journal of the Association for Computing Machinery*, 42:421–457.

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

Hertrampf, U. (1996). Acceptance by transformation monoids (with an application to local self reductions). Technical report, Fachbereich Informatik, Universität Stuttgart.

Hertrampf, U., Lautemann, C., Schwentick, T., Vollmer, H., and Wagner, K. W. (1993). On the power of polynomial time bit-reductions. In *Proceedings 8th Structure in Complexity Theory*, pages 200–207.

Immerman, N. (1987). Languages that capture complexity classes. *SIAM Journal on Computing*, 16:760–778.

Immerman, N. (1989). Expressibility and parallel complexity. *SIAM Journal on Computing*, 18:625–638.

Immernan, N. and Landau, S. (1995). The complexity of iterated multiplication. *Information and Computation*, 116:103–116.

Jenner, B., McKenzie, P., and Thérien, D. (1994). Logspace and logtime leaf languages. In *9th Annual Conference Structure in Complexity Theory*, pages 242–254.

Lindell, S. (1992). A purely logical characterization of circuit uniformity. In *Proceedings 7th Structure in Complexity Theory*, pages 185–192.

Lindström, P. (1966). First order predicate logic with generalized quantifiers. *Theoria*, 32:186–195.

Papadimitriou, C. H. (1994). *Computational Complexity*. Addison-Wesley.

Regan, K. and Vollmer, H. (1996). Gap-languages and log-time complexity classes. *Theoretical Computer Science*. To appear.

Ruzzo, W. L. (1981). On uniform circuit complexity. *Journal of Computer and Systems Sciences*, 21:365–383.

Sipser, M. (1983). Borel sets and circuit complexity. In *Proceedings of the 15th Symposium on Theory of Computing*, pages 61–69. ACM Press.

Stewart, I. A. (1991). Comparing the expressibility of languages formed using NP-complete operators. *Journal of Logic and Computation*, 1:305–330.

Stewart, I. A. (1992). Using the Hamilton path operator to capture NP. *Journal of Computer and System Sciences*, 45:127–151.

Toda, S. (1991). PP is as hard as the polynomial time hierarchy. *SIAM Journal on Computing*, 20:865–877.

Veith, H. (1996). Succinct representation, leaf languages, and projection reductions. In *Proceedings 10th Computational Complexity*, pages 118–126. IEEE Computer Society Press.

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

Wagner, K. W. (1986). The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23:325–356.

Wagner, K. W. and Wechsung, G. (1986). *Computational Complexity*. VEB Verlag der Wissenschaften, Berlin.