Definability on Finite Structures and the Existence of One-Way Functions

Erich Grädel
Mathematisches Institut
Universität Basel
Rheinsprung 21
CH-4051 Basel
graedel@urz.unibas.ch

Final version
January 1993

Abstract

Craig's interpolation theorem and Beth's definability theorem are classical results in model theory that fail when only finite structures are considered. Gurevich (*Toward Logic Tailored for Computational Complexity*, Springer Lecture Notes in Mathematics 1104 (1984) 175–216) has shown that for any logic that captures polynomial time, the analogues of these theorems (on finite structures) are equivalent to certain open statements in complexity theory. By results of Grollmann and Selman (*Complexity measures for public-key cryptosystems*, SIAM J. Computing **17** (1988), 309–335) these statements are false if and only if there exist certain kinds of one-way functions (for polynomial time). We extend these results and give direct proofs of the correspondance between one-way functions for a complexity class $\mathcal{C}$ and the definability and interpolation principles for any logic that captures $\mathcal{C}$.

1 Introduction

Intuitively a one-way function is a function which is easy to compute and hard to invert. The distinction between "easy" and "hard" is usually set by polynomial time computability. But even then there are several points that need clarification: What does it mean, that a *partial* function is computable in polynomial time (or any other complexity bound)? Does it mean that a Turing machine computes $f(x)$ in polynomial time on all x where f is defined and enters an infinite loop on (or rejects) all other x? This is the usual definition; it implies that the domain of f, denoted $D(f)$, is in P and that f can be extended to a total polynomial time computable function which coincides with f on $D(f)$. However, this is a rather restrictive definition because we impose conditions also on the elements for which f is not defined. A more liberal requirement is that there is a Turing machine which computes $f(x)$ in polynomial time for every $x \in D(f)$ (and we don't care what it does on inputs outside of $D(f)$). On the other side, there is also a more restrictive definition, which is sometimes used, requiring that also the image of f, denoted $I(f)$, is recognizable in polynomial time.

More generally, we can relate the complexity of computing a (partial) function in three different ways to decision problems. For an arbitrary (complexity) class $\mathcal{C}$ of decision problems, we thus obtain three different notions of $\mathcal{C}$-computability of a function. This will also give us the notions of a $\mathcal{C}$-one-way function and a strong, respectively weak, $\mathcal{C}$-one-way function.

The computation of (partial) functions and their complexity are related to logic, or more precisely, finite model theory via the notion of expressibility. For decision problems the relationship is well-known: a decision problem can be considered as a set L of finite structures of some fixed vocabulary, and L is expressed by a formula ψ iff the set of finite models of ψ is precisely L. In a similar way a formula $\psi(x_1, \ldots, x_k)$ with k free variables represents a function that assigns to every structure of appropriate vocabulary a k-ary relation. This relationship allows to translate model-theoretic statements into assertions in complexity theory (like the existence of one-way functions) and vice versa.

Two important results in model theory are Craig's interpolation theorem and Beth's definability theorem (see section 3); it is however known that these two theorems, like many other important results on first order logic, fail on finite structures [10]. Analogues of the interpolation and the definability theorem can be formulated for arbitrary logics and have indeed been investigated for almost every logic that might be of interest to model theorists (see [16]). But most of this research has focussed on arbitrary structures and bears little significance to the finite case. Gurevich has treated the interpolation principle, the definability principle and a weaker version thereof for logics that capture polynomial time; he proved that these principles are equivalent to open statements in complexity theory which are generally believed to be false: If $\mathcal{L}$ captures polynomial time then the interpolation principle for $\mathcal{L}$ holds if and only if any two disjoint set in NP are separated by a set in P (this would imply that NP $\cap$ Co-NP = P). The definability principle for $\mathcal{L}$ is equivalent to UP = P, where UP is the set of NP problems with unique witnesses, a class introduced by Valiant [18]; finally the weak definability principle corresponds to the equality UP $\cap$ Co-UP = P. Results of Grollmann and Selman [9] show that the converse of any of these three statements is equivalent to the existence of a certain kind of one-way function (for polynomial time).

In this paper we directly relate the interpolation principle, the definability principle and the weak definability principle (on finite structures) for any logic $\mathcal{L}$ with the existence of one-way functions for the class $\mathcal{C}$ that is captured by $\mathcal{L}$. The proof is purely model-theoretic and applies to every logic which is closed under first order operations.

2 Preliminaries

A *vocabulary* or *signature* is a set $\sigma = \{P_1, \ldots, P_r, x_1, \ldots, x_s\}$ of relation variables (each of which has some specified arity) and element variables. A σ*-structure* $\mathfrak{B}$ consists of a nonempty set $|\mathfrak{B}|$ (the universe), of predicates $P_1^{\mathfrak{B}}, \ldots, P_r^{\mathfrak{B}}$ over $|\mathfrak{B}|$ of the same arities as $P_1, \ldots, P_r$, and of constants $x_1^{\mathfrak{B}}, \ldots, x_s^{\mathfrak{B}} \in |\mathfrak{B}|$. The class of all σ-structures is denoted by $\mathrm{Str}(\sigma)$ and the class of all finite σ-structures by $\mathrm{Fin}(\sigma)$. If $\sigma \subseteq \sigma'$ and $\mathfrak{B}$ is a σ'-structure, then the σ*-reduct* $\mathfrak{B}|_\sigma$ is the σ-structure obtained by omitting from $\mathfrak{B}$ all predicates and constants that are not in σ. By default, structures in this paper are finite and have universe of the form $\{0, \ldots, n-1\}$ where $n \in \mathbb{N}$.

In this paper we are interested in the status of certain model theoretic principles for different logics when only finite structures are considered. These principles have first been formulated for first-order logic (on the class of arbitrary structures). Later model theorists became interested in other logics and investigated, how much of the model theory of first-order logic can be carried over to these more general logics. To do such investigations, it is useful to have a general notion of what constitutes a logic, so that the principles under study can be formulated in an abstract way. We will use the following definition of a logic, taken from [6]:

Definition 2.1 A *logic* (or logical system) consists of a pair $(\mathcal{L}, \models)$. $\mathcal{L}$ associates with every signature σ a set $\mathcal{L}(\sigma)$ whose elements are called $\mathcal{L}$-sentences of vocabulary σ. The semantic of $\mathcal{L}$-sentences is given by the binary relation $\models$ which is a subset of $\bigcup_\sigma(\mathrm{Str}(\sigma) \times \mathcal{L}(\sigma))$. If $\mathfrak{B} \models \psi$, we say that $\mathfrak{B}$ is a model of ψ. The following properties are required:

(i) If $\sigma \subseteq \sigma'$, then $\mathcal{L}(\sigma) \subseteq \mathcal{L}(\sigma')$.

(ii) (Isomorphism property) If $\mathfrak{A} \models \psi$ and $\mathfrak{A} \cong \mathfrak{B}$, then $\mathfrak{B} \models \psi$.

(iii) (Reduct property) If $\sigma \subseteq \sigma'$, $\psi \in \mathcal{L}(\sigma)$ and $\mathfrak{B}$ is a σ'-structure with $\mathfrak{B} \models \psi$, then also $\mathfrak{B}|_\sigma \models \psi$.

To enhance readability, we usually denote a logic just by the the set of its sentences, i.e. we write $\mathcal{L}$ rather than $(\mathcal{L}, \models)$.

Definition 2.1 gives a very general notion of a logic. For specific purposes, it is sometimes useful to impose additional conditions (see e.g. [6, 16, 11]). Here we will require that the logics satisfy certain closure properties on finite structures:

A logic $\mathcal{L}$ is closed under *positive first-order operations* (on finite structures) if, for any formulae $\psi, \varphi \in \mathcal{L}(\sigma)$, there exist formulae $(\psi \wedge \varphi)$, $(\psi \vee \varphi)$, $(\exists x)\psi$ and $(\forall x)\psi$ (where x is a first-order variable) with the appropriate semantic. For instance, the finite models of $(\exists x)\psi$ must be precisely the $(\sigma - \{x\})$-reducts of the finite models for ψ. A logic is *closed under first-order operations* if it is closed under positive first-order operations and also under negation: for every $\psi \in \mathcal{L}(\sigma)$ there exists a formula $(\neg\psi) \in \mathcal{L}(\sigma)$ such that, $\mathfrak{A} \models (\neg\psi)$ if and only if $\mathfrak{A} \not\models \psi$, for all finite structures $\mathfrak{A}$ of appropriate vocabulary. All logics that we consider in this paper are closed under positive first-order operations. However, there are interesting logics, that are not closed under negations or where this is open, such as e.g. Σ^1_1, the logic that captures NP. Note, that it is important that we require these properties to hold on finite structures; e.g. Σ^1_1 is known not to be closed under negation on arbitrary structures, but for finite structures, this is equivalent to the conjecture that NP $\neq$ Co-NP.

Definition 2.2 The vocabulary of a formula $\psi \in \mathcal{L}$ is the minimal σ such that $\psi \in \mathcal{L}(\sigma)$. It is denoted by $\mathrm{voc}(\psi)$.

Definition 2.3 For every formula ψ, we denote by $\mathrm{MOD}(\psi)$ the set of *finite models* of ψ. We say that ψ *expresses* a decision problem $L \subseteq \mathrm{Fin}(\sigma)$ if $L = \mathrm{MOD}(\psi)$. A logic $\mathcal{L}$ *captures* a class $\mathcal{C}$ of decision problems if $\mathcal{C} = \{\mathrm{MOD}(\psi) \mid \psi \in \mathcal{L}\}$.

Logical characterizations of this kind are known for most major complexity classes. Fagin's theorem states that Σ_1^1 (i.e. existential second order logic) captures precisely NP [7]; a straightforward extension shows that second order logic as a whole corresponds to the polynomial-time hierarchy [17]. For PTIME and lower complexity classes, the known logical characterizations require that the underlying structures are ordered (e.g. by a successor relation). Immerman and Vardi [12, 19] proved that the problems solvable in polynomial time are those that are expressible in the least fixpoint logic (FO + LFP), i.e. first order logic together with a least fixed point operator. Immerman systematically studied the problem of designing logics that capture complexity classes [12]–[14] and came up with logical descriptions for many other important classes. As in the case of fixpoint logic, most of these logical characterisations are obtained by augmenting the syntax of first order logic by operators, e.g. by various forms of transitive closure operators. In the absence of a linear ordering these logics can still express certain complete problems for the respective complexity classes, but they fail to express some very simple properties, such as "parity", i.e. the property that the cardinality of a given structure is even [4]. Alternative characterisations of complexity classes by fragments of second order logic were given in [8]. The most important results in this field are surveyed in [14, 11].

In this paper, we are interested in describing functions rather than decision problems. Therefore, instead of sentences, we need formulae with free variables.

Definition 2.4 In an abstract logic $\mathcal{L}$, as given by Definition 2.1, a *formula* $\psi(\bar{x})$ of vocabulary σ *with free variables* $x_1, \ldots, x_r$ (that do not occur in σ) is a sentence in $\mathcal{L}(\sigma \cup \{x_1, \ldots, x_r\})$. Interpreted on a fixed σ-structure $\mathfrak{B}$, such a formula defines a relation $\{\bar{x}^{\mathfrak{B}} \mid (\mathfrak{B}, \bar{x}^{\mathfrak{B}}) \models \psi(\bar{x})\} \subseteq |\mathfrak{B}|^r$. But on different structures $\psi(\bar{x})$ may define of course different relations. The object defined by $\psi(\bar{x})$ is therefore a function f_ψ that assigns to a given structure $\mathfrak{B}$ a relation $f_\psi(\mathfrak{B}) \subseteq |\mathfrak{B}|^r$. Moreover, this function is *invariant under automorphisms*: If h is an automorphism of $(\mathfrak{B})$, then $f_\psi(h(\mathfrak{B})) = h(f_\psi(\mathfrak{B}))$. Such a function is called a query.

Definition 2.5 A *query* of arity $r \geq 0$ and vocabulary σ is an automorphism-invariant, possibly partial function that maps structures $\mathfrak{B} \in \mathrm{Fin}(\sigma)$ to relations $f(\mathfrak{B}) \subseteq |\mathfrak{B}|^r$. In the case where $r = 0$, the query is Boolean, i.e. a decision problem. The *domain* of f is denoted by $D(f)$. In the literature, queries are also called *global relations* [11].

Definition 2.6 Let f be a query of arity $r > 0$ and vocabulary σ. A formula $\psi(x_1, \ldots, x_r)$ (in any logic $\mathcal{L}$) *explicitely defines* f if $f(\mathfrak{B}) = \{\bar{x} \mid \mathfrak{B} \models \psi(\bar{x})\}$ for every σ-structure $\mathfrak{B} \in D(f)$.

Note that that this definition imposes no condition on the logical definability of the domain of f or on the truth-values of $\psi(\bar{x})$ on elements $\bar{x} \notin D(f)$.

Definition 2.7 Let $\mathcal{C}$ be any complexity class. By $\Sigma_1^1\mathcal{C}$ we denote the class of existential projections of sets in $\mathcal{C}$, i.e. $L \in \Sigma_1^1\mathcal{C}$ iff there exists a set $L' \in \mathcal{C}$ such that

$$L = \{\mathfrak{B} \mid (\exists \bar{P})(\mathfrak{B}, \bar{P}) \in L'\}.$$

The class $\Pi_1^1\mathcal{C}$ is defined in an analogous way with universal instead of existential quantifiers. Finally $\Delta_1^1\mathcal{C}$ is $\Sigma_1^1\mathcal{C} \cap \Pi_1^1\mathcal{C}$. A similar operation applies to a logic $\mathcal{L}$: Define a new logic $\Sigma^1_1\mathcal{L}$ as the set of formulae $(\exists P_1)\cdots(\exists P_r)\varphi$ where $P_1,\ldots,P_r$ are predicate symbols and φ is a formula from $\mathcal{L}$. If $\mathcal{L}$ captures $\mathcal{C}$ then clearly $\Sigma^1_1\mathcal{L}$ captures $\Sigma^1_1\mathcal{C}$; the converse is of course in general not true. A property is said to be $\Delta^1_1\mathcal{L}$-definable iff it is expressible by a formula from $\Sigma^1_1\mathcal{L}$ and also by a formula from $\Pi^1_1\mathcal{L}$.

We will sometimes use the following notation: If ψ is a formula containing the k-ary predicate P and φ is a formula with k free first-order variables, then

$$[P\bar{x} \equiv \varphi(\bar{x})]\psi$$

denotes the formula obtained by replacing every occurrence of P in ψ by φ. (The expression $[P\bar{x} \equiv \varphi(\bar{x})]$ binds variables just as a quantifier does, so it goes before the formula to which it applies.)

3 Interpolation and definability

Craig's interpolation theorem [5] states that for every valid implication $\psi \to \varphi$ of first order formulae, there exists a first order formula θ such that $\mathrm{voc}(\theta) \subseteq \mathrm{voc}(\psi) \cap \mathrm{voc}(\varphi)$ and such that the implications $\psi \to \theta$ and $\theta \to \varphi$ are valid. It is known that this theorem fails if validity is replaced by validity on finite structures (see [10]).

Let $\mathcal{L}$ be any logic. We can formulate the following

> *Interpolation principle for $\mathcal{L}$:* For every pair of formulae ψ, φ of $\mathcal{L}$ such that the implication $\psi \to \varphi$ is valid on finite structures, there is a formula $\theta \in \mathcal{L}$, called an *interpolant*, with $\mathrm{voc}(\theta) \subseteq \mathrm{voc}(\psi)\cap\mathrm{voc}(\varphi)$ such that the implications $\psi \to \theta$ and $\theta \to \varphi$ are valid on finite structures.

Remark. This formulation of the interpolation principle is slightly different from the one given in [10]: We just require that the formulae ψ, φ and θ be in $\mathcal{L}$, but not necessarily the implications $\psi \to \varphi$, $\psi \to \theta$ and $\theta \to \varphi$. This may make a difference for logics that are not closed under negation.

We are interested in the interpolation principle for various logics that are related to complexity classes. It trivially holds for second order logic and also for its Σ_i- and Π_i-fragments (for $i > 0$). Indeed, let $\psi \to \varphi$ be a valid implication with $\mathrm{voc}(\psi) = \{\bar{P}, \bar{R}\}$ and $\mathrm{voc}(\varphi) = \{\bar{P}, \bar{S}\}$. Then both $(\exists\bar{R})\psi$ and $(\forall\bar{S})\varphi$ are interpolants for $\psi \to \varphi$.

However, the interpolation principle does probably not hold for any logic that captures P, since it would imply that P = NP $\cap$ Co-NP [10]. More generally we have the following, well-known result (see [16]):

Proposition 3.1 *If the interpolation principle holds for $\mathcal{L}$ then every property which is $\Delta^1_1\mathcal{L}$-definable is already expressible in $\mathcal{L}$.*

PROOF. Consider a property in $\Delta^1_1\mathcal{L}$, defined by formulae $(\exists\bar{P})\psi$ and $(\forall\bar{R})\varphi$ with $\psi, \varphi \in \mathcal{L}$. We can assume that $\bar{P}$ and $\bar{R}$ are disjoint (otherwise we rename predicates). Then $\psi \to \varphi$ is valid on finite structures, so there exists an interpolant θ. It follows that also the implications $(\exists\bar{P})\psi \to \theta$ and $\theta \to (\forall\bar{R})\varphi$ are valid on finite structures, so the implications are in fact equivalences. ∎

Another consequence of Craig's interpolation theorem is Beth's definability theorem which concerns the notion of implicit definability:

Definition 3.2 Let f be a query of vocabulary σ. A formula $\psi(P)$ of vocabulary $\sigma \cup \{P\}$ *implicitely defines* f if $D(f) = \{\mathfrak{B} \mid \mathfrak{B} \models (\exists P)\psi\}$ and for all structures $\mathfrak{B} \in D(f)$ and all relations P

$$(\mathfrak{B}, P) \models \psi \iff f(\mathfrak{B}) = P.$$

In particular there exists for every $\mathfrak{B}$ at most one P such that $(\mathfrak{B}, P)$ is a model for ψ.

Beth's definability theorem [3] states the equivalence of implicit and explicit definability in first order logic, i.e. for every query f which is implicitely defined by a first-order formula $\psi(P)$ of vocabulary $\sigma \cup \{P\}$, there exists another first-order formula $\varphi(\bar{x})$ which defines the same query explicitely (according to Definition 2.6). Note that the query may be partial, i.e. the formula $(\exists P)\psi(P)$ need not be true in every σ-structure. Like Craig's interpolation theorem, Beth's definabilty theorem fails in the finite case [10]. Again we can formulate for an arbitrary logic $\mathcal{L}$ the

> *Definability principle:* Every (partial) query on $\mathrm{Fin}(\sigma)$ which is implicitely definable in $\mathcal{L}$ is also explicitely definable in $\mathcal{L}$.

An often used variant of the definability principle is the weak definability principle which restricts attention to total queries:

> *Weak definability principle:* Every totally defined query on $\mathrm{Fin}(\sigma)$ which is implicitely definable in $\mathcal{L}$ is also explicitely definable in $\mathcal{L}$.

Gurevich proved that for any logic capturing P, the definability principle is equivalent to UP = P, and the weak definability principle is equivalent to UP $\cap$ Co-UP = P.

4 Complexity classes and one-way functions

Most complexity classes are defined for decision problems. To talk about the complexity of, say, a partial function $f : \{0,1\}^* \to \{0,1\}^*$ one can define special complexity classes for such objects, and/or consider the complexity of a related decision problem.

Definition 4.1 Let $\mathcal{S}_k$ be the set of finite structures whose vocabulary consists of a single k-ary relation. To simplify notation we denote a structure $(\{0,\ldots,n-1\}, P) \in \mathcal{S}_k$ simply by its relation P.

Any decision problems (at least if it is encodable as a set of words over a finite alphabet) can be represented as a set of finite structures. Similarly we can encode a (partial) function as a query on $\mathcal{S}_k$: A query on $\mathcal{S}_k$ of arity m is a partial function $f : \mathcal{S}_k \to \mathcal{S}_m$ that preserves the universe. For simplicity we will only consider functions with $f(\varnothing) = \varnothing$.

Definition 4.2 For all $k, m \in \mathbb{N}$, we denote by $\mathcal{G}_{k,m}$ be the set of all functions $f : \mathcal{S}_k \to \mathcal{S}_m$ which preserve the universe and which map the empty relation to the empty relation. Let $\mathcal{G} = \bigcup_{k,m} \mathcal{G}_{k,m}$.

Remark. To represent a (partial) function $f : \{0,1\}^* \to \{0,1\}^*$ in $\mathcal{G}$ it must be required that the length of $f(x)$ is uniformly bounded by a polynomial in the length of x. If also inverse functions are to be considered, then the functions should be polynomially honest. However, since we consider directly the functions in $\mathcal{G}$ we need not be concerned with these issues.

In the sequel, $\mathcal{L}$ is always a logic that is closed under positive first-order operations, and $\mathcal{C}$ is a class of decision problems (think of $\mathcal{C}$ as a complexity class) that is captured by $\mathcal{L}$. It is common practice to associate with a partial function $f : \{0,1\}^* \to \{0,1\}^*$ the decision problem

$$L_f := \{(x,i) \mid x \in D(f) \text{ and the } i\text{th bit of } f(x) \text{ is } 1\}.$$

We associate with a query $f \in \mathcal{G}$ an analogous decision problem:

Definition 4.3 We say that $f \in \mathcal{G}$ is *$\mathcal{C}$-computable* if

$$L_f := \{(P,\bar{a}) \mid P \in D(f) \wedge \bar{a} \in f(P)\} \in \mathcal{C}.$$

Equivalently, this means that there is a formula $\psi(P,\bar{x})$ in $\mathcal{L}$ such that $L_f = \mathrm{MOD}(\psi)$.

In case that $\mathcal{C}$ is PTIME, this is one of the three reasonable notions (in fact the most usual one) of polynomial-time computability of partial functions, that were mentioned at the very beginning of this paper: There is polynomial time algorithm, which given an instance x for f and a "bit number", determines, whether x belongs to the domain of f, and, if yes, whether the corresponding bit of $f(x)$ is one. Since there are only polynomially many bits to consider, this means that $f(x)$ can be computed in polynomial time.

A less restrictive notion of $\mathcal{C}$-computability, which does not impose that inputs outside the domain of f must be rejected, is given by

Definition 4.4 A function $f \in \mathcal{G}$ is *weakly $\mathcal{C}$-computable* if and only if

(i) f is explicitely definable in $\mathcal{L}$;

(ii) there is a formula $\psi \in \Sigma^1_1\mathcal{L}$ such that $D(f) = \mathrm{MOD}(\psi)$.

Note that this definition is purely logical. To give an equivalent definition in terms of $\mathcal{C}$ rather that $\mathcal{L}$ we introduce the set

$$\tilde{L}_f := \{(P,\bar{a}) \mid P \in D(f) \Longrightarrow \bar{a} \in f(P)\}$$

which, for functions $f : \{0,1\}^* \to \{0,1\}^*$ corresponds to the decision problem $\{(x,i) \mid x \in D(f) \Longrightarrow \text{ the } i\text{th bit of } f(x) \text{ is } 1\}$.

Lemma 4.5 *A function $f \in \mathcal{G}$ is weakly $\mathcal{C}$-computable if and only if*

(i) there is a set $L \in \mathcal{C}$ such that $L_f \subseteq L \subseteq \tilde{L}_f$;

(ii) $D(f) \in \Sigma^1_1\mathcal{C}$.

PROOF. It is clear that *(ii)* is equivalent to condition *(ii)* of Definition 4.4. Explicit definability of f means, that there exists a formula $\varphi(P, \bar{x})$ in $\mathcal{L}$, such that for all $P \in D(f)$,

$$\{\bar{x} \mid (P, \bar{x}) \models \varphi\} = f(P).$$

But this is equivalent to $L_f \subseteq \mathrm{MOD}(\varphi) \subseteq \tilde{L}_f$, so condition *(i)* of the Lemma is equivalent to condition *(i)* of Definition 4.4. ∎

The requirement that the domain of f be expressible in $\Sigma_1^1\mathcal{L}$ may not seem natural at first sight. However, it is necessary to exclude functions that are difficult just because their domain is a very complicated, possibly even non-recursive, set. When applied to complexity classes between AC^0 and polynomial time this means that the domain of f must be in NP. Note that a function is weakly PTIME-computable if there exists a nondeterministic polynomially time-bounded Turing machine which accepts precisely the inputs $x \in D(f)$ and produces on every accepting path the same output $f(x)$.

On the other side there also is an even more restrictive notion of $\mathcal{C}$-computablity

Definition 4.6 A function $f \in \mathcal{G}$ is *strongly $\mathcal{C}$-computable* if f is $\mathcal{C}$-computable and the image $I(f)$ of f is included in $\mathcal{C}$.

Definition 4.7 We say that f is *(strongly, weakly) $\mathcal{C}$-invertible* if f is one-one and its inverse is (strongly, weakly) $\mathcal{C}$-computable. A function is a *(strong, weak) one-way function for $\mathcal{C}$* if f is one-one, (strongly, weakly) $\mathcal{C}$-computable, but not (strongly, weakly) $\mathcal{C}$-invertible.

The following lemma shows that to prove invertibility, only the explicit definability of the inverse function has to be verified.

Lemma 4.8 *Let $\mathcal{L}$ be a logic which is closed under first order operations and which captures $\mathcal{C}$. If $f \in \mathcal{G}$ is one-one and $\mathcal{C}$-computable and if f^{-1} is explicitely definable in $\mathcal{L}$, then both f and f^{-1} are strongly $\mathcal{C}$-computable.*

PROOF. Let $f : P \mapsto R$ be a one-one, $\mathcal{C}$-computable function in $\mathcal{G}$ and $g = f^{-1}$. There exists a formula $\psi_f(P, \bar{x})$ of $\mathcal{L}$ such that $\mathrm{MOD}(\psi_f) = L_f$, and a formula $\varphi_g(R, \bar{y}) \in \mathcal{L}$ which explicitely defines g according to Definition 2.6. It is clear that $D(f)$ is expressible by the formula

$$\alpha(P) \equiv (\exists \bar{x})\psi_f(P, \bar{x}) \vee (\forall \bar{y})\neg P\bar{y}.$$

It therefore suffices to show that $D(g) = I(f)$ is in $\mathcal{C}$; to do this we show that $D(g)$ is expressible in $\mathcal{L}$. We construct a formula $\delta(R) \in \mathcal{L}$ with free predicate R whose finite models are precisely the elements of $D(g)$.

$$\delta(R) \equiv (\forall \bar{x})\Big(R\bar{x} \leftrightarrow [P\bar{y} \equiv \varphi_g(R, \bar{y})]\psi_f(P, \bar{x})\Big).$$

Since $\mathcal{L}$ is closed under first order operations, $\delta(R)$ belongs to $\mathcal{L}$. We claim that $D(g) = \mathrm{MOD}(\delta)$.

Suppose that $R_0 \in D(g)$. Then the relation $P_0 := \{\bar{y} \mid (R_0, \bar{y}) \models \varphi_g\}$ is $g(R_0)$ and therefore belongs to $D(f)$. Thus $\psi_f(P_0, \bar{x})$ states that $\bar{x} \in f(P_0) = R_0$; it follows that R_0 is a model of δ.

Conversely, assume that $R_0 \models \delta$. If R_0 is empty, then it belongs to $D(g)$ by definition. If R_0 is non-empty there exists an element $\bar{x}$ such that $\psi_f(P_0, \bar{x})$ is true for the relation $P_0 = \{\bar{y} \mid (R_0, \bar{y}) \models \varphi_g\}$. This implies that $P_0 \in D(f)$. Morover it follows that $f(P_0) = R_0$ and therefore $R_0 \in I(f) = D(g)$. ■

In particular, a function which is $\mathcal{C}$ computable and weakly $\mathcal{C}$-invertible is strongly $\mathcal{C}$-computable and strongly $\mathcal{C}$-invertible.

Remark. These notions of one-way functions come from structural complexity theory (see e.g. [9]). For applications in cryptology they are not really adequate, simply because worst-case complexity is not the appropriate concept. In cryptology, a one-way function must be hard to invert almost everywhere (or at least on a substantial fraction of the inputs). Otherwise, a public key cryptosystem based on such a function might allow an efficient algorithm that would successfully decode most cyphertexts; such a cryptosystem is of course useless. We will not elaborate on this point; it might be interesting to extend our approach to this more restrictive notion of one-way functions.

5 One-way functions and the interpolation principle

From now on $\mathcal{L}$ always denotes a logic which is closed under first order operations, and $\mathcal{C}$ is the class of problems captured by $\mathcal{L}$.

Theorem 5.1 *The interpolation principle for $\mathcal{L}$ holds if and only if every injective function which is weakly $\mathcal{C}$-computable, is also weakly $\mathcal{C}$-invertible.*

Proof. Suppose that the interpolation principle is true for $\mathcal{L}$ and let $f \in \mathcal{G}_{k,m}$ be one-one and weakly $\mathcal{C}$-computable. Then, by Definition 4.4, there exists a formula $\varphi_f(P, \bar{x})$ which explicitely defines f; moreover there exists a formula $\delta(P, Q)$ such that $P \in D(f)$ iff $P \models (\exists Q)\delta$. Now construct the formula

$$\beta(P, R) \equiv (\forall \bar{x})(R\bar{x} \leftrightarrow \psi_f(P, \bar{x})).$$

For $P \in D(f)$ this formula says that $R = f(P)$.

Note that $D(f^{-1})$ is expressible in $\Sigma_1^1\mathcal{L}$ by the formula $(\exists P)(\exists Q)\big(\delta(P, Q) \wedge \beta(P, R)\big)$. It thus remains to be shown that f^{-1} is explicitely definable in $\mathcal{L}$.

Consider the implication

$$\Big(\delta(P, Q) \wedge \beta(P, R) \wedge P\bar{y}\Big) \rightarrow \Big(\big(\delta(P', Q') \wedge \beta(P', R)\big) \rightarrow P'\bar{y}\Big)$$

If the antecedent of this implication holds for P, Q, R and $\bar{y}$, then $R = f(P)$ and $\bar{y} \in P$. The conclusion says that $R = f(P')$ implies $\bar{y} \in P'$. Since f is one-one, this implication is valid. By the interpolation principle, there exist an interpolant $\theta(R, \bar{y})$. We claim that $\theta(R, \bar{y})$ is an explicit definition of f^{-1}. Indeed, assume that R_0 is in the domain of f^{-1}; there exists a unique P_0 with $f(P_0) = R_0$. For this P_0, $\beta(P_0, R_0)$ holds, and there also exists a Q_0 such that $\delta(P_0, Q_0)$ is satisfied. Thus, for $R_0 \in D(f^{-1})$ fixed, both sides of the implication become equivalent to $P_0\bar{y}$, i.e. to $\bar{y} \in f^{-1}(R_0)$. It follows that $P_0 = \{\bar{y} \mid (R_0, \bar{y}) \models \theta\}$. Thus f is $\mathcal{C}$-invertible.

Conversely, assume that every weakly $\mathcal{C}$-computable function is weakly $\mathcal{C}$-invertible and let $\psi \to \varphi$ be a valid implication; without loss of generality we may assume that $\mathrm{voc}(\psi) = \{P, \bar{A}\}$ and $\mathrm{voc}(\varphi) = \{P, \bar{B}\}$, i.e. that the common vocabulary of ψ and φ consists of a single predicate P. Furthermore we assume that if P is the empty relation, then $\psi(P, \bar{A})$ and $\psi(P, \bar{B})$ are true for all $\bar{A}$ and all $\bar{B}$ (if not, then replace ψ and φ by their disjunction with $(\forall \bar{x})\neg P\bar{x}$). Let R be a new predicate with $arity(R) = 2 \cdot arity(P)$. For every relation P let

$$\begin{aligned} R_1 &:= \{(\bar{x}, \bar{y}) \mid \bar{x} \in P\} \\ R_2 &:= \{(\bar{x}, \bar{x}) \mid \bar{x} \in P\} \end{aligned}$$

Let $f : \mathcal{S}_k \to \mathcal{S}_{2k}$ be defined by

$$f(P) = \begin{cases} R_1 & \text{if } P \models (\exists \bar{A})\psi \\ R_2 & \text{if } P \models (\exists \bar{B})\neg\varphi \\ \text{undefined} & \text{otherwise} \end{cases}$$

Since $\psi \to \varphi$, this mapping is well-defined and obviously one-one. We claim that f^{-1} is weakly $\mathcal{C}$-computable. Indeed f^{-1} is explicitely defined by $\varphi(R, \bar{x}) \equiv R\bar{x}\bar{x}$. Furthermore, the domain $D(f^{-1})$ is expressed by the formula $\delta_1(R) \vee \delta_2(R)$ where

$$\begin{aligned} \delta_1(R) &\equiv (\forall \bar{x})(\forall \bar{y})(R\bar{x}\bar{y} \leftrightarrow R\bar{x}\bar{x}) \wedge (\exists \bar{A})[P\bar{x} \equiv R\bar{x}\bar{x}]\psi \\ \delta_2(R) &\equiv (\forall \bar{x})(\forall \bar{y})(R\bar{x}\bar{y} \leftrightarrow \bar{x} = \bar{y}) \wedge (\exists \bar{B})[P\bar{x} \equiv R\bar{x}\bar{x}]\neg\varphi. \end{aligned}$$

Clearly $\delta_1(R) \vee \delta_2(R)$ belongs to $\Sigma_1^1\mathcal{L}$. By assumption, f is also weakly $\mathcal{C}$-computable. Thus, there is a formula $\psi_f(P, \bar{x}, \bar{y})$ which explicitely defines f; i.e. for every $P \in D(f)$, $f(P) = \{(\bar{x}, \bar{y}) \mid (P, \bar{x}, \bar{y}) \models \psi_f\}$.

The desired interpolant for $\psi \to \varphi$ is the formula

$$\theta(P) \equiv (\forall \bar{x})(\forall \bar{y})(\psi_f(P, \bar{x}, \bar{x}) \leftrightarrow P\bar{x}).$$

Indeed, assume that $(P, \bar{A}) \models \psi$; then $f(P) = R_1$ which implies that $P \models \theta$. On the other side, if $(P, \bar{B}) \models \neg\varphi$, then $f(P) = R_2$ and therefore $P \models \neg\theta$. Thus $\psi \to \theta$ and $\theta \to \varphi$ are valid for all finite structures. This proves that the interpolation principle for $\mathcal{L}$ is true. ■

Corollary 5.2 *There exist weak one-way functions for $\mathcal{C}$ if and only if the interpolation principle fails for $\mathcal{L}$.*

6 One-way functions and definability principles

Theorem 6.1 *The definability principle holds for $\mathcal{L}$ if and only if every $\mathcal{C}$-computable function is $\mathcal{C}$-invertible.*

PROOF. First we assume that the definability principle holds fo $\mathcal{L}$. Let $f : P \mapsto R$ be a one-one, $\mathcal{C}$-computable function in $\mathcal{G}$ and let $g = f^{-1}$. Then there exists a formula $\psi_f(P, \bar{x})$ with $\mathrm{MOD}(\psi_f) = L_f$. Let $\varepsilon(P, R)$ be a formula that expresses that P and R are either both empty or both nonempty (recall that $f(\varnothing) = \varnothing$ for all $f \in \mathcal{G}$). The formula

$$\beta(P, R) \equiv \varepsilon(P, R) \vee (\forall \bar{x})(R\bar{x} \leftrightarrow \psi_f(P, \bar{x}))$$

says that $f(P) = R$; since f is one-one $\beta(P, R)$ can be considered as an implicit definition of g. By the definability principle there exist an explicit definition $\varphi_g(R, \bar{y})$ of g, so it follows from Lemma 4.8 that g is $\mathcal{C}$-computable.

Conversely, assume that every $\mathcal{C}$-computable function is $\mathcal{C}$-invertible. It is well-known that structures of any finite vocabulary can be encoded by structures from $\mathcal{S}_k$ for any $k \geq 2$. It thus suffices to prove the definability principle for queries in $\mathcal{G}$. So let $\alpha(P, R)$ be an implicit definition of the function $f : P \mapsto R$ in $\mathcal{G}_{k,m}$. Consider the function $g : \mathcal{S}_{k+m} \to \mathcal{S}_k$ whose domain is $D(g) = \{T = P \times R \mid R = f(P)\}$ and which maps $T = P \times R$ to P. This function is defined by the formula

$$\begin{aligned}\psi_g(T, \bar{x}) \quad \equiv \quad & (\forall \bar{u})(\forall \bar{u}')(\forall \bar{v})(\forall \bar{v}')(T\bar{u}\bar{v} \wedge T\bar{u}'\bar{v}' \to T\bar{u}\bar{v}') \wedge \\ & [P\bar{u} \equiv (\exists \bar{v})T\bar{u}\bar{v}][R\bar{v} \equiv (\exists \bar{u})T\bar{u}\bar{v}]\alpha(P, R) \wedge (\exists \bar{y})T\bar{x}\bar{y}.\end{aligned}$$

Thus g is $\mathcal{C}$-computable and clearly one-one. Its inverse $h := g^{-1}$ has domain $D(f)$ and maps P to $P \times f(P)$. So $f(P) = \{\bar{y} \mid (\exists \bar{x})(\bar{x}, \bar{y}) \in h(P)\}$. By assumption h is $\mathcal{C}$-computable and, in particular, explicitely definable by a formula $\psi_h(P, \bar{x}, \bar{y})$ in $\mathcal{L}$. The desired explicit definition of f is then

$$\varphi_f(P, \bar{y}) \equiv (\exists \bar{x})\psi_h(P, \bar{x}, \bar{y}).$$

■

Theorem 6.2 *The weak definability principle holds for $\mathcal{L}$ if and only if every strongly $\mathcal{C}$-computable function is $\mathcal{C}$-invertible (and therefore also strongly $\mathcal{C}$-invertible).*

Proof. Assume that the weak definability principle holds for $\mathcal{L}$. Let $f \in \mathcal{G}_{k,m}$ be a strongly $\mathcal{C}$-computable, injective function; thus there exist formulae $\delta(P)$, $\psi_f(P, \bar{y})$ and $\eta(R)$ such that $\mathrm{MOD}(\delta) = D(f)$, $\mathrm{MOD}(\psi_f) = L_f$ and $\mathrm{MOD}(\eta) = I(f)$.

Define the following total function $g : \mathcal{S}_m \to \mathcal{S}_{k+2m}$:

$$g(R) := \begin{cases} \{(\bar{x}, \bar{y}, \bar{y}) \mid \bar{x} \in f^{-1}(R), \bar{y} \in R\} & \text{if } R \in I(f) \\ \{(\bar{x}, \bar{y}, \bar{z}) \mid \bar{z} \in R\} & \text{if } R \notin I(f) \end{cases}$$

Obviously g is in $\mathcal{G}$, one-one and defined everywhere. Using δ, ψ_f and η, it is a matter of routine to construct a formula $\beta(R, T)$ which implicitely defines g. By the weak definability principle there exists an explicit definition $\psi_g(R, \bar{x}, \bar{y}, \bar{z})$ of g. Then the desired explicit definition of f^{-1} is

$$\varphi(R, \bar{x}) \equiv \eta(R) \wedge (\exists \bar{y})\psi_g(R, \bar{x}, \bar{y}, \bar{y}).$$

The converse follows with essentially the same proof as for the previous theorem. If $\alpha(P, R)$ is an implicit definition of a total function $f \in \mathcal{G}_{k,m}$, then the function g mapping $T = P \times f(P)$ to P is $\mathcal{C}$-computable and therefore strongly $\mathcal{C}$-computable (because f is total). By the weak definibility theorem there exists an explicit definition of g^{-1} and therefore also of f. ■

Corollary 6.3 *(i) There exist one-way functions for $\mathcal{C}$ if and only if the definability principle fails for $\mathcal{L}$.*

(ii) There exist strong one-way functions for $\mathcal{C}$ if and only if the weak definability principle fails for $\mathcal{L}$.

7 Further results

To conclude we briefly survey some related results.

Proposition 7.1 *The interpolation principle fails for every logic $\mathcal{L}$ which has at least the power of first order logic, but does not express all problems of* NP $\cap$ Co-NP.

PROOF. If $\mathcal{L}$ contains first order logic then every problem in NP $\cap$ Co-NP is $\Delta_1^1\mathcal{L}$-definable. Therefore this result is an immediate consequence of Proposition 3.1. ∎

In particular, the interpolation principle fails for the infinitary logic $L^{\omega}_{\infty\omega}$ [2], and for all logics and query languages between first-order logic and $L^{\omega}_{\infty\omega}$ such as least fixpoint logic [4, 12], partial fixpoint logic (or, equivalently, first-order logic together with while-looping [4, 1, 19]), transitive closure logic [13], Stratified Datalog and most other query languages (in the absence of a linear ordering).

On finite structures implicit definitions seem to be stronger than explicit ones. It is therefore interesting to determine the expressive power of implicit definitions (see [15]). For this purpose it is useful to generalize the notion of an implicit definition.

Definition 7.2 Let f be a query of vocabulary σ. A formula $\psi(P_0, \dots, P_r)$ with $\mathrm{voc}(\psi) = \sigma \cup \{P_0, \dots, P_r\}$ is a *generalized implicit definition of f* if for every σ-structure $\mathfrak{B}$ there is at most one tuple $P_0, \dots, P_r$ of relations with $(\mathfrak{B}, P_0, \dots, P_r) \models \psi$ and if

$$(\mathfrak{B}, P_0, \dots, P_r) \models \psi \quad \Longrightarrow \quad P_0 = f(\mathfrak{B}).$$

It is tempting to use this notion to define a *generalized definability principle* by weakening the antecedent; it states that generalized implicit definability in $\mathcal{L}$ implies explicit definability in $\mathcal{L}$. However, this generalized principle is not stronger than the original one.

Proposition 7.3 *Let $\mathcal{L}$ be a logic which is closed under substitution. Then the definability principle for $\mathcal{L}$ implies the generalized definability principle for $\mathcal{L}$.*

We omit the proof which is easy. The following theorem, due to Kolaitis [15], shows that implicit definability is a rather robust notion.

Theorem 7.4 *On finite structures, generalized implicit definability in* FO + LFP *implies generalized implicit definability in first order logic.*

It is easy to see that this theorem does not survive when generalized implicit definability is replaced by implicit definability. In fact the transitive closure is not implicitely definable (in the restricted sense) in first order logic, even in the presence of a linear ordering. The following is a slightly modified and strengthened form of Theorem 3.11 in Kolaitis' paper [15]:

Corollary 7.5 *Let $\mathcal{C} \subseteq$* P*. If there exists a (strong) $\mathcal{C}$-one-way function then there also exists a (strong) $\mathcal{C}$-one-way functions which admits a generalized implicit first order definition.*

As mentioned before, most logical characterizations of complexity classes require the presence of a linear ordering. For any logic $\mathcal{L}$ we denote by $\mathcal{L}_{\leq}$ the logic that is obtained by adding the relation $\leq$ as a logical constant (like $=$) to be interpreted in the usual way (recall that all our universes are initial subsets of the natural numbers).

Theorem 7.6 *A query f admits a generalized implicit definition in* $\mathrm{FO}_{\leq}$ *if and only if f is* UP*-computable.*

This result is essentially also due to Kolaitis, although it appears in different form in his paper. Kolaitis considers implicit definitions of total queries. In the presence of a linear ordering these coincide with UP $\cap$ Co-UP instead of UP.

Lemma 7.7 *Let $\mathcal{L}$ be a logic that has at least the expressive power of universal first order formulae. Then, on finite structures, the definability principle for $\mathcal{L}$ implies the definability principle for $\mathcal{L}_{\leq}$.*

PROOF. A linear ordering is axiomatizable by a universal first order formula $\alpha(\leq)$. Suppose that we have an implicit definition $\psi(P)$ in $\mathcal{L}_{\leq}$ of a query on ordered structures from $\mathrm{Fin}(\sigma)$. This means that for every ordered σ-structure $\mathfrak{B}$ there is at most one predicate P such that $(\mathfrak{B}, P) \models \psi$. If we replace every occurrence of $\leq$ in ψ by a new binary predicate S, and call this formula $\psi(P, S)$ then $\alpha(S) \wedge \psi(P, S)$ is an implicit definition on $F(\sigma \cup \{S\})$. By the definability principle there is an equivalent explicit definition $\varphi(\bar{x}) \in \mathcal{L}$. Replacing S by $\leq$ gives the desired explicit definition in $\mathcal{L}_{\leq}$. ■

Corollary 7.8 *Let $\mathcal{L}$ be an extension of first order logic such that $\mathcal{L}_{\leq}$ captures a proper subclass of* UP. *Then the definability principle fails for $\mathcal{L}$ and $\mathcal{L}_{\leq}$.*

PROOF. For every problem $L \in$ UP there is a partial function f with domain L that admits a generalized implicit definition $\psi(R_1, \ldots, R_t)$ of f in $\mathrm{FO}_{\leq}$, hence also in $\mathcal{L}_{\leq}$. If the definability principle would hold for $\mathcal{L}_{\leq}$ then all the implicitely defined relations $R_1, \ldots, R_t$ were explicitely definable by formulae $\varphi_i(\bar{x})$ from $\mathcal{L}_{\leq}$. But then L would be expressible in $\mathcal{L}_{\leq}$ by the formula

$$[R_1(\bar{x}) \equiv \varphi_1(\bar{x})] \cdots [R_t(\bar{x}) \equiv \varphi_t(\bar{x})]\psi$$

which contradicts the assumption that $\mathcal{L}$ captures a proper subset of UP. Thus the definability principle fails for $\mathcal{L}_{\leq}$ and therefore also for $\mathcal{L}$. ■

Acknowledgement. I'm grateful to Heinz-Dieter Ebbinghaus for his comments on an earlier version of this paper.

References

[1] S. Abiteboul and V. Vianu. *Fixpoint Extensions of First-Order Logic and Datalog-like Languages*, Proceedings of 4th IEEE Symposium on Logic in Computer Science (1989), 71–79.

[2] J. Barwise, *On Moschovakis closure ordinals*, J. Symbolic Logic **42** (1977), 292–296.

[3] E. Beth, *On Padoa's method in the theory of definition*, Indag. Math **15** (1953), 330–339.

[4] A. Chandra and D. Harel, *Structure and Complexity of Relational Queries*, J. Comp. Syst. Sciences **25** (1982), 99–128.

[5] W. Craig, *Three Uses of the Herbrand-Gentzen Theorem in Relating Model Theory and Proof Theory*, J. Symbolic Logic **22** (1957), 250-268.

[6] H.-D. Ebbinghaus, J. Flum and W. Thomas, *Mathematical Logic*, Springer-Verlag, 1984.

[7] R. Fagin, *Generalized First-Order Spectra and Polynomial-Time Recognizable Sets*, SIAM-AMS Proc. **7** (1974), 43–73.

[8] E. Grädel, *Capturing Complexity Classes by Fragments of Second Order Logic*, Theoretical Computer Science **101** (1992), 35–57.

[9] J. Grollmann and A. Selman, *Complexity measures for public-key cryptosystems*, SIAM J. Computing **17** (1988), 309–335.

[10] Y. Gurevich, *Toward Logic Tailored for Computational Complexity*, in: "Computation and Proof Theory". Proceedings of Logic Colloquium 1983, Lecture Notes in Mathematics Nr. 1104, Springer 1984, 175–216.

[11] Y. Gurevich, *Logic and the Challenge of Computer Science*, in: "Trends in Theoretical Computer Science" (E. Börger, Ed.), Computer Science Press (1988), 1–57.

[12] N. Immerman, *Relational Queries Computable in Polynomial Time*, Inf. and Control **68** (1986), 86–104.

[13] N. Immerman, *Languages that Capture Complexity Classes*, SIAM J. Comput. **16** (1987), 760–778.

[14] N. Immerman, *Descriptive and Computational Complexity*, in: J. Hartmanis (Ed.), Computational Complexity Theory, Proc. of AMS Symposia in Appl. Math. **38** (1989), 75–91.

[15] Ph. Kolaitis, *Implicit Definability on Finite Structures and Unambigous Computations*, Preliminary Report, University of California, Santa Cruz, November 1989. See also: Proceedings of 5th IEEE Symp. on Logic in Computer Science (1990), 168–180.

[16] *Model-Theoretic Logics,* edited by J. Barwise and S. Feferman, Springer 1985.

[17] L. Stockmeyer, *The polynomial-time hierarchy*, Theoretical Computer Science **3** (1977), 1-22.

[18] L. Valiant, *Relative complexity of checking and evaluating*, Information Processing **5** (1976), 20–23.

[19] M. Vardi, *Complexity of Relational Query Languages*, Proc. of 14th STOC (1982), 137–146.