Journal of Computer and System Sciences 80 (2014) 1152–1162

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

A characterization of definability of second-order generalized quantifiers with applications to non-definability

Juha Kontinen [a,*], Jakub Szymanik [b]

[a] Department of Mathematics and Statistics, University of Helsinki, Finland
[b] Institute for Logic, Language, and Computation, University of Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 14 January 2012
Received in revised form 21 February 2013
Accepted 3 February 2014
Available online 4 April 2014

Keywords:
Second-order generalized quantifiers
Definability
Majority quantifier
Second-order logic
Collective quantification

ABSTRACT

We study definability of second-order generalized quantifiers. We show that the question whether a second-order generalized quantifier $\mathcal{Q}_1$ is definable in terms of another quantifier $\mathcal{Q}_2$, the base logic being monadic second-order logic, reduces to the question if a quantifier $\mathcal{Q}_1^\star$ is definable in $\mathrm{FO}(\mathcal{Q}_2^\star, <, +, \times)$ for certain first-order quantifiers $\mathcal{Q}_1^\star$ and $\mathcal{Q}_2^\star$. We use our characterization to show new definability and non-definability results for second-order generalized quantifiers. We also show that the monadic second-order majority quantifier Most[1] is not definable in second-order logic.

1. Introduction

The notion of generalized quantifier goes back to Mostowski [1] and Lindström [2]. Generalized quantifiers were first mainly studied in the framework of model theory. The study of generalized quantifiers extended to the context of finite model theory via applications to descriptive complexity theory. We refer to [3] and [4] for surveys of first-order generalized quantifiers in finite model theory. Generalized quantifiers have been also extensively studied in the formal semantics of natural language (see [5] for a survey).

The study of second-order generalized quantifiers is a relatively new and unexplored area in finite model theory. On the other hand, second-order logic (SO) and its many fragments have been studied extensively starting from Fagin's characterization of NP in terms of existential second-order logic [6]. Second-order generalized quantifiers were first studied in the context of finite structures by Burtschick and Vollmer [7]. Shortly after, Andersson [8] studied the expressive power of families of second-order generalized quantifiers determined by the syntactic types of quantifiers. In [9–11] Kontinen studied definability questions of second-order generalized quantifiers. In the case of first-order quantifiers, definability of a quantifier Q in a logic $\mathcal{L}$ means that the class of structures, used to interpret Q, is axiomatizable in $\mathcal{L}$. In the second-order case, the analogous concept of definability was formulated in [9,10]. In this article, we give a computationally motivated characterization for the notion of definability of second-order generalized quantifiers. Burtschick and Vollmer [7] noticed that second-order generalized quantifiers can be used to logically characterize complexity classes defined in terms of so-called *Leaf Languages*. The leaf languages approach in computational complexity theory, introduced by Bovet, Crescenzi, and Silvestri [12], is a unifying approach to define complexity classes. The central idea behind this approach is to generalize the conditions under which, e.g., a Turing machine or an automaton accepts its input. Many complexity classes can be defined in this context in terms of suitable leaf languages. On the other hand, a complexity

* Corresponding author.

http://dx.doi.org/10.1016/j.jcss.2014.04.007
0022-0000/

class defined in terms of a leaf language B can be under certain conditions characterized logically by a logic of the form:

$$\mathcal{Q}_B\mathrm{FO},$$

where $\mathcal{Q}_B$ is a second-order generalized quantifier corresponding to the language B. In the context of leaf languages, polynomial time non-deterministic Turing machines can be sometimes replaced by non-deterministic finite automata (so-called finite leaf automata) without a significant decrease in complexity [13]. Galota and Vollmer [14] showed that complexity classes defined by finite leaf automata can be logically characterized in terms of monadic second-order generalized quantifiers. This result nicely extends the well known [15–17] characterization of regular languages in terms of monadic second-order logic (MSO).

The definability theory of second-order generalized quantifiers has some similarities and differences compared to that of first-order generalized quantifiers. For example, it was observed in [9] that the binary second-order existential quantifier cannot be defined in terms of any monadic second-order generalized quantifiers. This result is in contrast with the fact (a corollary of a result of Andersson [8]) that all classes of finite first-order structures are already definable in terms of monadic second-order generalized quantifiers.

In this paper we prove a general result characterizing the question when a quantifier $\mathcal{Q}$ is definable in $\mathrm{MSO}(\mathcal{Q}', +)$, where $+$ denotes the built-in addition relation. We assume the built-in addition in order to unleash the expressive power embodied by MSO. Recall that, while MSO corresponds to regular languages over strings, $\mathrm{MSO}(+)$ corresponds to the linear fragment of the polynomial hierarchy (LINH) on strings [18]. Some of our results can be generalized to the case where the base logic is full second-order logic instead of $\mathrm{MSO}(+)$. Our characterization is based on the idea connecting oracle separation results with lower bound results for small constant depth circuits, see e.g., [19–23]. We show that a second-order generalized quantifier $\mathcal{Q}_1$ is definable in the logic $\mathrm{MSO}(\mathcal{Q}_2, +)$ iff for certain first-order encodings $\mathcal{Q}_i^\star$ of $\mathcal{Q}_i$, $\mathcal{Q}_1^\star$ is definable in $\mathrm{FO}(\mathcal{Q}_2^\star, +, \times)$. It is worth noting that the latter condition implies that $\mathcal{Q}_1^\star$ is AC^0 (Turing) reducible to $\mathcal{Q}_2^\star$. We use our characterization to show new definability and non-definability results for second-order generalized quantifiers. In particular, we show that the monadic second-order majority quantifier Most^1 is not definable in second-order logic. This answers the question left open in [24] (see also [25]), where second-order generalized quantifiers were used to model collective quantification in natural language, for example:

1. Most girls gathered.
2. All soldiers surrounded the Alamo.

The common strategy in formalizing collective quantification has been to define the meanings of collective determiners, quantifying over collections, using certain type-shifting operations. These operations, i.e., lifts, define the collective interpretations of determiners systematically from the standard meanings of quantifiers (see, e.g., [26,27]). In [24] we show that all these lifts are definable in second-order logic. In this paper we prove that some collective quantifiers (second-order generalized quantifiers) are not definable in second-order logic. Therefore, there is no second-order definable lift expressing their collective meaning. This is clearly a restriction of the type-shifting approach. One possible alternative would be to use second-order generalized quantifiers in the study of collective semantics, as we already proposed in [24]. However, as it follows from this paper the computational complexity of such approach is excessive and hence it may not be a plausible model of collective quantification in natural language (see [28–30] for a discussion of computational restrictions in natural language semantics). Hence, it may be wise to turn in the direction of another well-known way of studying collective quantification in natural language, the many-sorted (algebraic) tradition (see [31]). Another linguistic interpretation of our results might be that computational complexity restricts the expressive power of everyday language (see [32]). Namely, even though natural language can in principle realize collective quantifiers non-definable in second-order logic, its everyday fragment does not contain such constructions due to their high complexity.

2. Preliminaries

In this article all structures are assumed to be finite. The universe of a structure $\mathfrak{A}$ is denoted by A. Without loss of generality, we may assume that A is always of the form $\{0, \ldots, m\}$ for some $m \in \mathbb{N}$. For a logic $\mathcal{L}$, the set of τ-formulas of $\mathcal{L}$ is denoted by $\mathcal{L}[\tau]$. If ϕ is a τ-sentence, then the class of τ-models of ϕ is denoted by $\mathrm{Mod}(\phi)$. A class K of τ-models is said to be axiomatizable in a logic $\mathcal{L}$, if $K = \mathrm{Mod}(\phi)$ for some sentence $\phi \in \mathcal{L}[\tau]$. For logics $\mathcal{L}$ and $\mathcal{L}'$, we write $\mathcal{L} \leqslant \mathcal{L}'$, if for every τ and every sentence $\phi \in \mathcal{L}[\tau]$ there is a sentence $\psi \in \mathcal{L}'[\tau]$ such that $\mathrm{Mod}(\phi) = \mathrm{Mod}(\psi)$. The set of natural numbers is denoted by $\mathbb{N}$ and $\mathbb{N}^*$ denotes the set $\mathbb{N} \setminus \{0\}$.

Sometimes we assume that our structures (and logics) are equipped with auxiliary built-in relations. In addition to the built-in ordering $<$, which is interpreted naturally, we also use the ternary relations $+$ and $\times$. The relations $+$ and $\times$ are defined as

$$+(i, j, k) \quad \Leftrightarrow \quad i + j = k,$$
$$\times(i, j, k) \quad \Leftrightarrow \quad i \times j = k.$$