Information and Computation 247 (2016) 290–312

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Introduction to clarithmetic II

Giorgi Japaridze

Department of Computing Sciences, Villanova University, 800 Lancaster Avenue, Villanova, PA 19085, USA

A R T I C L E I N F O

Article history:
Received 12 August 2011
Received in revised form 17 May 2014
Available online 10 February 2016

Keywords:
Computability logic
Game semantics
Peano arithmetic
Constructive theories
Interactive computation
Bounded arithmetic
Implicit computational complexity

A B S T R A C T

The earlier paper "Introduction to clarithmetic I" constructed an axiomatic system of arithmetic based on computability logic, and proved its soundness and extensional completeness with respect to polynomial time computability. The present paper elaborates three additional sound and complete systems in the same style and sense: one for polynomial space computability, one for elementary recursive time (and/or space) computability, and one for primitive recursive time (and/or space) computability.

1. Introduction

Being a continuation of [9], this article fully and heavily relies on the terminology, notation, conventions and technical results of its predecessor, with which the reader is assumed to be well familiar (the good news, however, is that, [9], in turn, is self-contained).

Remember, from [9], the system **CLA4** of arithmetic, both semantically and syntactically based on computability logic (CoL). Its language was that of Peano arithmetic (**PA**) augmented with the choice conjunction $\sqcap$, choice disjunction $\sqcup$, choice universal quantifier $\bigsqcap$ and choice existential quantifier $\bigsqcup$. On top of the standard Peano axioms, **CLA4** had two extra-Peano axioms: $\bigsqcap x \bigsqcup y(y = x+1)$ and $\bigsqcap x \bigsqcup y(y = 2x)$, one saying that the function $x+1$ is computable, and the other saying the same about the function $2x$. The only logical rule of **CLA4** was Logical Consequence (LC), meaning that the logical basis for the system was the (sound and complete) fragment **CL12** of CoL. And the only nonlogical rule of inference was the induction rule

$$\frac{F(0) \qquad F(x) \to F(2x) \qquad F(x) \to F(2x+1)}{F(x)},$$

with $F(x)$ — that is, its choice quantifiers $\bigsqcap, \bigsqcup$ — required to be polynomially bounded. The system was proven in [9] to be sound and extensionally (representationally) complete with respect to polynomial time computability.

The present paper constructs three new **CL12**-based systems: **CLA5**, **CLA6**, **CLA7** and proves their soundness and extensional completeness with respect to polynomial space computability, elementary recursive time (and/or space) computability, and primitive recursive time (and/or space) computability, respectively. While **CLA4** was already simple enough, the

E-mail address: giorgi.japaridze@villanova.edu.
URL: http://www.csc.villanova.edu/~japaridz.

http://dx.doi.org/10.1016/j.ic.2016.02.002

above three systems are even more so. All of them only need $\sqcap x \sqcup y(y = x + 1)$ as a single extra-Peano axiom. As before, the only logical rule is LC. And the induction rule (the only nonlogical rule) of each of these systems is

$$\frac{F(0) \qquad F(x) \to F(x+1)}{F(x)}.$$

The three systems differ from each other only in what (if any) conditions are imposed on the formula $F(x)$ of induction. In **CLA5**, as in **CLA4**, $F(x)$ is required to be polynomially bounded. **CLA6** relaxes this requirement and allows $F(x)$ to be an exponentially bounded formula. **CLA7** takes this trend towards relaxation to an extreme and imposes no restrictions on $F(x)$ whatsoever. This way, unlike **CLA4**, **CLA5** and **CLA6**, theory **CLA7** is no longer in the realm of bounded arithmetics.

The simplicity and elegance of these systems is additional evidence for the naturalness and productiveness of the idea of basing complexity-oriented systems and bounded arithmetic in particular on CoL instead of classical logic, even if one is only concerned with functions rather than the more general class of all interactive computational problems. In [1], achieving representational completeness with respect to polynomial space computable functions required considering a second-order extension of classical-logic-based bounded arithmetic (similarly in [2] for certain other complexity classes). In our case, on the other hand, a transition from polynomial time (**CLA4**) to polynomial space (**CLA5**) in remarkably smooth with no need for any changes in the underlying language or logic, and with only minimal syntactic changes in the nonlogical part (induction rule) of the system. Among the virtues of CoL is that, as a logic, it remains the same regardless of for what purposes (polynomial time computability, polynomial space computability, computability-in-principle, ...) it is used. CoL does not have *variations*, but rather has various (conservative) *fragments*,[1] depending on what part of its otherwise very expressive language is considered. The fragment dealt with in the present paper, as well as in its predecessor [9], as well as in its even earlier predecessors [8,10], is the same: logic **CL12**.

1.1. Technical notes

All terminology and notation not redefined in this paper has the same meaning as in [9]. And all of our old conventions from [9] extend to the present context as well.

Additionally we agree that a "**sentence**" always means a sentence (closed formula) of the language of **CLA4**. Similarly for "**formula**", unless otherwise specified or suggested by the context.

The definition of a polynomially bounded formula given in Section 11 of [9] contained a minor technical error. The correct formulation, on which we shall subsequently rely and which was really meant throughout [9], is as follows. We say that a formula F is **polynomially bounded** iff every subformula $\sqcap x G(x)$ (resp. $\sqcup x G(x)$) of F has the form $\sqcap x(S(x) \to H(x))$ (resp. $\sqcup x(S(x) \wedge H(x))$), where $S(x)$ is a polynomial sizebound for x none of whose free variables is bound by $\forall$ or $\exists$ within F.

In the context of a given play (computation branch) of an HPM $\mathcal{M}$, by the **spacecost** of a given clock cycle c we shall mean the number of cells ever visited by the work-tape head of $\mathcal{M}$ by time c. We extend the usage of this term from clock cycles to the corresponding configurations as well.

As in the preceding paragraph, we will be using the informal term "**play**" mostly in reference to a computation branch of a given machine, but occasionally it should rather be understood as the run spelled by such a branch. The meaning will usually be clear from the context.

2. CLA5, a theory of polynomial space computability

The **language** of theory **CLA5** is the same as that of **CLA4** — that is, it is an extension of the language of **PA** through the additional binary connectives $\sqcap, \sqcup$ and quantifiers $\sqcap, \sqcup$. And the axiomatization of **CLA5** is obtained from that of **CLA4** by deleting Axiom 9 (which is now redundant) and replacing the **CLA4**-Induction rule by the following rule, which we call **CLA5-Induction**:

$$\frac{\sqcap\big(F(0)\big) \qquad \sqcap\big(F(x) \to F(x')\big)}{\sqcap\big(F(x)\big)},$$

where $F(x)$ is any polynomially bounded formula. Here we shall say that $\sqcap\big(F(0)\big)$ is the **basis** of induction, and $\sqcap\big(F(x) \to F(x')\big)$ is the **inductive step**.

To summarize, the nonlogical axioms of **CLA5** are those of **PA** (Axioms 1–7) plus one single additional axiom $\sqcap x \sqcup y(y = x')$ (Axiom 8). There are no logical axioms. The only logical inference rule is Logical Consequence (LC) as defined in Section 10 of [9], and the only nonlogical inference rule is **CLA5**-Induction.

The following fact establishes that the old Axiom 9 of **CLA4** would indeed be redundant in **CLA5**:

[1] Including what has been termed "intuitionistic computability logic" (studied in [4–6]), contrary to what this name may suggest. Unlike, say, intuitionistic linear logic, which is indeed a variation of (classical) linear logic, intuitionistic computability logic is merely a conservative fragment of CoL, obtained by restricting its logical vocabulary to the choice operators and the ultimate reduction operator.