数理解析研究所講究録 1442 巻 2005 年 68-75



# Logical Characterizations of LOGSPACE

Satoru Kuroda (黒田　覚)
Gunma Prefectural Women's University (群馬県立女子大学)

## 1 Introduction

Bounded arithmetic is a branch of mathematical logic which characterize various classes of computational complexity by fragments of arithmetical theories. On the other hand, descriptive complexity gives another logical method to characterize complexity classes. However, until recently, no actual connection was known between these two branches of mathematical logic.

S. Cook and A. Kolokolova [2] gave an elegant method to define second order systems of bounded arithmetic which utilizes the descriptive characterization of PTIME and NL. Their method uses the fragments of second order logic which was used by E. Grädel to characterize those complexity classes.

In this paper we will give a fragment of second order logic which characterizes LOGSPACE over finite ordered structures. This result may be used to obtain a second order system of bounded arithmetic similar to Cook and Kolokolova system for NL and P.

Descriptive characterizations for LOGSPACE is obtained by Immerman and Grädel. The former one is by introducing deterministic transitive closure (DTC) to first order logic, while the latter is a fragment of second order Krom logic. Our result is similar to the latter one, but the formula we use is more natural.

## 2 Descriptive complexity and LOGSPACE

### 2.1 Basic Notions

We will briefly review basic definitions of descriptive complexity. For details, readers are encouraged to refer Immerman [4]

A signature is a finite set of constant symbols and relation symbols. We

do not include function symbols in signature but rather add thier graphs as relation symbols. Signatures are denoted by $\sigma$, $\tau$ and so on.

Let $\sigma$ be a signature. A $\sigma$-structure is a tuple $\mathcal{A} = \langle A, \{R^{\mathcal{A}}\}_{R\in\sigma}, \{c^{\mathcal{A}}\}_{c\in\sigma}\rangle$ where $A$ is a possibly finite set called *universe*, $R^{\mathcal{A}}$ is a relation on $A$ for each relation symbol $R \in \sigma$ and $c^{\mathcal{A}} \in A$ for each constant symbol $c \in \sigma$. We will denote the number of elements in $A$ by $|A$.

**Example 1** *1. Let $E$ be a binary relation and $\sigma = \{E\}$. Then the $\sigma$-structure $\mathcal{A} = \langle\{0, 1, \ldots, n-1\}, E^{\mathcal{A}}\rangle$ represents a directed graph.*

*2. The signature $\sigma = \{\min, \max, S\}$ is called a successor signature, where* min *and* max *are constant symbols denoting minimal and maximal elements of the universe respectively, and $S$ is a relation symbol denoting the successor relation.*

*3. $\sigma = \{\min, \max, \leq\}$ is called an ordered structure where* min *and* max *are as above and $\leq$ is the usual order relation.*

*4. $\sigma = \{\min, \max, \leq, +, \times\}$ is called an arithmetical structure where $+$ and $\times$ are graphs of addition and multiplication respectively.*

Lower case letters $x, y, z, \ldots$ denote first order variables and upper case letters $X, Y, Z, \ldots$ denote second order variables. Since we do not have function symbols in signatures, terms are either first order variables or constants. Formulae are built up from atomic formulae by applying the logical connectives $\wedge$, $\vee$, $\neg$, $\forall$, $\exists$. A *logic* is a set of formulae.

In order to make strict connections between logics and complexity classes, it must be made clear how we regard finite structures as binary strings and vice versa.

A structure $\mathcal{A} = \{A, \bar{R}^{\mathcal{A}}, \bar{c}^{\mathcal{A}}\}$ can be regarded as a binary string by coding each relation $R^{\mathcal{A}}$ by a binary string $w_{R^{\mathcal{A}}}$ of length $n^k$ such that

$$(x_1, \ldots, x_n) \in R^{\mathcal{A}} \Leftrightarrow \text{the corresponding bit of } w_{R^{\mathcal{A}}} \text{ is 1.}$$

Similarly, each constant symbol $c^{\mathcal{A}}$ is coded by a binary string $w_{c^{\mathcal{A}}}$ of length $n$ such that

$$c^{\mathcal{A}} = 1 \Leftrightarrow j\text{th bit of } w_{c^{\mathcal{A}}} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

Finally, the code of the whole structure $\mathcal{A}$ is the concatenation of codes for relations and constants defined as above.

Conversely, a binary string $w$ of length $n$ can be represented by a structure $\langle\{0, 1, \ldots, n-1\}, \min, \max, \leq\rangle$ by introducing a unary relation $P$ such that

$$P(i) \Leftrightarrow i\text{th bit of } w \text{ is 1.}$$

Now we a ready to state what it means that a logic corresponds to a complexity class.

**Definition 1** *Let $\mathcal{L}$ be a logic, $\mathcal{C}$ be a complexity class and $\mathcal{D}$ be a set of finite structures. We say that $\mathcal{L}$ captures $\mathcal{C}$ over $\mathcal{D}$ if the following conditions hold:*

1. *For all $\varphi \in \mathcal{L}$ and $A \in \mathcal{D}$, the model checking problem $\mathcal{A} \models \varphi$ is decidable in $\mathcal{C}$.*
2. *For each set $C \in \mathcal{C}$ there exists $\varphi \in \mathcal{L}$ such that*

$$\forall \mathcal{A} \in \mathcal{D}(\mathcal{A} \in \mathcal{C} \Leftrightarrow \mathcal{A} \models \varphi).$$

Some relations between logics and complexity classes are well-known.

**Definition 2** *FO is the set of first order formulae, that is, all quantifiers are first order quantifiers. SO is the set of second order formulae. SO$\exists$ is the set of formulae of the form $\exists \bar{P} \varphi(\bar{P})$ where $\varphi \in FO$.*

**Theorem 1** *Over ordered structures, FO captures the class uniform $AC^0$, that is, the class of sets computable by constant depth, polynomial size boolean circuits of unbounded fan-in.*

**Theorem 2 (Fagin's Theorem)** *SO captures the polynomial hierarchy over all structures. SO$\exists$ captures the class NP over all structures.*

It is not known whether there exists a logic which captures P or smaller complexity classes over all structures. However, several results are known about the relations between logic and complexity classes over ordered structures. So in the following, will concentrate on ordered structures.

There are two ways to capture complexity classes such as P, NL, L and so on, namely, extending first order logic by operators or generalized quantifiers, and restricting second order logic.

## 2.2 Extending first order logic

By introducing additional operators to first order logic, various complexity classes below P are characterized. Below we will give some examples.

**Definition 3** *Let $P$ be a predicate symbol not in the signature $\sigma$ and $\varphi(\bar{x})$ be a formula of the signature $\sigma \cup \{P\}$ with only positive occurrences of $P$ and with free variables $\bar{x} = x_1, \ldots, x_n$. For each $\sigma$-structure $\mathcal{A}$, $\varphi$ defines a mapping*

$$\varphi_{\mathcal{A}} \; : \; P \mapsto \{\bar{a} : (\mathcal{A}, P) \models \varphi(\bar{a})\}.$$

*Since $P$ appears only positively, this mapping is monotone. So $\varphi_{\mathcal{A}}$ has a least fixed point defined by*

$$P^0 := \emptyset,$$
$$P^{j+1} := \varphi_{\mathcal{A}}(P^j),$$
$$P^\omega := \bigcup_{j\in\omega} \varphi_{\mathcal{A}}(P^j).$$

*Now we introduce the least fixed point operator $LFP$ such that $\mathcal{A} \models [LFP_{P,\bar{x}}\varphi](\bar{a})$ if and only if $\bar{a}$ is contained in the least fixed point of $\varphi$ on the structure $\mathcal{A}$.*

*$FO + LFP$ is defined as the first order logic with a positive least fixed point operator $[LFP_{P,\bar{x}}\varphi]$.*

**Theorem 3 (Immerman [5])** *$FO + LFP$ captures the class $P$ over ordered structures.*

Another prominent example is the connection between transitive closure and logarithmic space.

**Definition 4** *Let $\varphi(\bar{x},\bar{y})$ be a formula with free variables $\bar{x} = x_1,\ldots,x_k$ and $\bar{y} = y_1,\ldots,y_k$. Then we define $[TC_{\bar{x},\bar{y}}\varphi(\bar{x},\bar{y})](\bar{u},\bar{v})$ to be a formula such that $\mathcal{A} \models [TC_{\bar{x},\bar{y}}\varphi(\bar{x},\bar{y})](\bar{u},\bar{v})$ if and only if there exists $\bar{c_0},\ldots,\bar{c_n} \in \mathcal{A}$ such that $\bar{c_0} = \bar{a}$, $\bar{c_n} = \bar{b}$ and $\mathcal{A} \models \varphi(\bar{c_i},\bar{c_{i+1}})$ for all $i < n$.*

*$FO + TC$ is the first order logic extended by the operator $TC$.*

**Theorem 4 (Immerman [5])** *$FO + TC$ captures the class of nondeterminintic logarithmic space computable sets $NL$ over ordered structures.*

There is also a deterministic version of $TC$.

**Definition 5** *The deterministic transitive closure operator $DTC$ is defined by*

$$[DTC_{\bar{x},\bar{y}}\varphi(\bar{x},\bar{y})] \equiv [TC_{\bar{x},\bar{y}}(\varphi(\bar{x},\bar{y}) \wedge \forall\bar{z}(\varphi(\bar{x},\bar{z}) \rightarrow \bar{y} = \bar{z})].$$

**Theorem 5 (Immerman [5])** *$FO + DTC$ captures the class of logarithmic space computable sets $NL$ over ordered structures.*

## 2.3 Fragments of second order logic

Second order logic can also be used to capture complexity classes. The following results are due to Grädel [3].

**Definition 6** *A second order formula is SO-Horn if it is of the form*

$$Q_1P_1\cdots Q_kP_k\forall\bar{y}\varphi(P_1,\ldots,P_k,\bar{y})$$

*where each $Q_1, \ldots, Q_k$ is either $\forall$ or $\exists$ and $\varphi$ is a quantifier free Horn formula with respect to $P_1, \ldots, P_k$, that is, it is in conjunctive norman form such that in each clause there is at most one positive occurrence of second order variable.*

*A formula is $(SO\exists)$-Horn if all second order quantifiers are existential.*

**Proposition 1** *SO-Horn collapses to $(SO\exists)$-Horn, that is, for all SO-Horn formula $\varphi$ there exists a $(SO\exists)$-Horn formula $\psi$ such that for any structure $\mathcal{A}$, $\mathcal{A} \models \varphi \leftrightarrow \psi$.*

**Theorem 6 (Grädel [3])** *SO-Horn captures P over ordered structures.*

**Definition 7** *A second order formula is SO-Krom if it is of the form*

$$Q_1 P_1 \cdots Q_k P_k \forall \bar{y} \varphi(P_1, \ldots, P_k, \bar{y})$$

*where each $Q_1, \ldots, Q_k$ is either $\forall$ or $\exists$ and $\varphi$ is a quantifier free Krom formula with respect to $P_1, \ldots, P_k$, that is, it is in conjunctive norman form such that in each clause there are at most two occurrences of second order variables.*

*A formula is $(SO\exists)$-Krom if all second order quantifiers are existential.*

**Proposition 2** *SO-Krom collapses to $(SO\exists)$-Krom, that is, for all SO-Krom formula $\varphi$ there exists a $(SO\exists)$-Krom formula $\psi$ such that for any structure $\mathcal{A}$, $\mathcal{A} \models \varphi \leftrightarrow \psi$.*

**Theorem 7 (Grädel [3])** *SO-Krom captures NL over ordered structures.*

## 2.4 A second order logic for LOGSPACE

Now we will state and prove our main result of this section. That is, we will present a fragment of second order logic which captures the class LOGSPACE.

Grädel's second order logics *SO*-Horn and *SO*-Krom make use of satisfiability problems which are complete for P and NL respectively. Analogously, we will use a satifiability problem which is complete for L to build a second order logic capturing L.

Johannsenn [6] showed that the following satisfiability problem is complete for L.

**Definition 8** *A propositional formula is $CNF(2)$ if it is in conjunctive normal form and each propositional variable occurs at most twice.*

$$SAT(2) := \{\varphi \in CNF(2) : \varphi \text{ is satisfiable}\}.$$

**Theorem 8 (Johannsen [6])** *$SAT(2)$ is complete for L.*

Using this result, we can define a fragment of second order logic as follows:

**Definition 9** *$(SO\exists)$-$CNF(2)$ is the set of formulae of the form*

$$\exists P_1 \cdots \exists P_k \forall \bar{y} \varphi(P_1, \ldots, P_k, \bar{y})$$

*where $\varphi(P_1, \ldots, P_k, \bar{y})$ is $CNF(2)$ with respect to $P_1, \ldots, P_k$, that is, a quantifier free formula in conjunctive normal form such that each $P_i$ appears at most twice.*

First we will show that the model checking problem for $(SO\exists)$-$CNF(2)$ formula is in L.

**Theorem 9** *Let $\varphi \in (SO\exists)$-$CNF(2)$. Then the set of finite models of $\varphi$ is in L.*

(Proof). Let $\varphi \equiv \exists P_1 \cdots P_k \forall \bar{y} \varphi_0(P_1, \ldots, P_k, \bar{y})$ where $\varphi_0$ is a $CNF(2)$ formula with respect to $P_1, \ldots, P_k$ and let $\mathcal{A}$ be a finite structure. First eliminate the universal quantifier $\forall \bar{y}$ by transforming $\forall \bar{y} \varphi_0(\bar{P}, \bar{y})$ to a conjunction over all elements of $\mathcal{A}$. This yields a formula of the form

$$\exists P_0 \cdots \exists P_k \varphi_1(P_1, \ldots, P_k)$$

where $\varphi_1$ is a CNF formula such that each clause is a disjunction of $P_i$'s and first order atomic formulae. Evaluate eahc atomic formula in $\mathcal{A}$. If it is evaluated to true then eliminate all clauses which contain it. Otherwise eliminate all occurrences of the atomic formula.

Now the original formula is transformed into a formula of the form

$$\exists P_1 \cdots P_k \varphi_2(P_1, \ldots, P_k)$$

where $\varphi_2$ is considered as a propositional $CNF(2)$ formula. Thus checking whether $\mathcal{A} \models \exists P_1 \cdots \exists P_k \varphi_2(P_1, \ldots, P_k)$ is the satisfiability problem of $CNF(2)$ formula. So by Theorem 8 it is in L.

Next we will show that $(SO\exists)$-$CNF(2)$ is strong enought to express all LOGSPACE relations. To do this, we will use an operator representing a complete problem for LOGSPACE.

**Proposition 3 (Johannsen [6])** *The following tree-freeness (TF) problem is complete for LOGSPACE:*

> *TF : Given an undirected graph $G$, does every connected component in $G$ contain a cycle?*

**Definition 10** *Let $\varphi(\bar{x}, \bar{y})$ be a formula with free parameters as shown, where $\bar{x}$ and $\bar{y}$ are $k$-tuples of variables. We define the tree-freeness operator $\mathrm{T}F_{\bar{x},\bar{y}}\varphi$ in such a way that $\mathcal{A} \models \mathrm{T}F_{\bar{x},\bar{y}}\varphi$ if and only if the undirected graph whose edge relation is defined by $\varphi$ on the structure $\mathcal{A}$ is tree-free.*

*$\mathrm{T}F(FO)$ is the logic which consists of formulae of the form $\mathrm{T}F_{\bar{x},\bar{y}}\varphi$ where $\varphi$ is a quantifier free formula.*

Then the following holds:

**Theorem 10** *$\mathrm{T}F(FO)$ captures LOGSPACE over ordered structures.*

(Proof). This is an immediate consequence of Proposition 3.

Now it suffices to show that $(SO\exists)$-$CNF(2)$ formulae can express the tree-freeness operator $\mathrm{T}F$.

**Theorem 11** *For all quantifier free formula $\varphi$ there exists a $(SO\exists)$-$CNF(2)$ formula $\psi$ such that for all ordered structure $\mathcal{A}$, $\mathcal{A} \models \mathrm{T}F_{\bar{x},\bar{y}}\varphi \leftrightarrow \psi$.*

(Proof). Given and undirected graph $G = (V, E)$ we will construct a $CNF(2)$ formula as follows: First introduce a variable $P_e$ for each edge $e \in E$. For each vertex $v \in V$ we construct a clause $C_v$ which contains one literal for each edge $e$ which is incident upon $v$. If $e$ connects $v$ to a vertex with greater number then $C_v$ contains $x_e$ and $C_v$ contains $\bar{x}_e$ otherwise.

Now it is obvious that This formula is $CNF(2)$ and it is also readily proved that $G$ is tree-free if and only if this formula is satisfiable. Thus we have proved the theorem.

# 3 Further research

Combining the result in this paper and the work of Cook and Kolokolova [2], we can construct a second order bounded arithmetic which corresponds to LOGSPACE. Several other theory for LOGSPACE are obtained by D. Zambella [7] and Clote and Takeuti [1]. Using our result, we can construct a theory whose only nontrivial axiom is the comprehension scheme for $(SO\exists)$-$CNF(2)$ formulae, which is simpler than other systems.

It is known that LOGSPACE is equivalent to the class of predicates which are computable by polynomial size branching programs. This fact has some applications in logical approaches to the class LOGSPACE. Cook defined the first propositional proof system which corresponds to LOGSPACE using the branching program based characterization of the class.

Concering bounded arithmetic, we can define a boolean algebra based on the set of polynomial size branching programs. So the boolean valued method of Takeuti and Yasumoto [8] is also available for LOGSPACE.

# References

[1] Clote, P. and G. Takeuti, First order bounded arithmetic and small boolean circuit complexity classes. In Feasible Mathematics, II, Birkhäuser, (1995).

[2] Cook, S. and A. Kolokolova, A second order theory for NL.

[3] Grädel, E., Capturing complexity classes by fragments of second-order logic. Theoretical Computer Science, 101 (1992), pp.35–57.

[4] Immerman, N., Descriptive Complexity. Graduate Texts in Computer Science, 1998, Springer.

[5] Immerman, N., Languages that capture complexity classes, SIAM J. Compt., 16 (1987), pp. 760–778.

[6] Johannsen, J., Satisfiability problems complete for deterministic logarithmic space. In: V.Diekert and M. Habib eds. STACS 2004, Lecture Notes in Computer Science, 2996 (2004), pp. 317–325.

[7] Zambella, D., Notes on polynomially bounded arithmetic. The Journal of Symbolic Logic, 61(1996), pp. 942–966.

[8] Takeuti, G. and M. Yasumoto, Forcing on Bounded Arithmetic, In. Gödel '96, A K Peters, (1996), pp. 120–138.