On computational complexity of successor theory with unary transitive closure

Sergey M Dudakov
CS Dept., Tver State University, Tver, 170100, Russia
E-mail: sergeydudakov@yandex.ru

Abstract. The present article concerns the theory of the successor function with the unary transitive closure (TC) operator. This theory is equivalent to the TC-theory of discrete linear order with respect to TC-definability. We prove that any decision algorithm for this theory has at least hyperexponential computational complexity. The latter is much higher than the complexity of the same theories without the TC-operator.

1. Introduction
The decidability of mathematical theories and the computational complexity of decision algorithms are among the most focused problems in the mathematical logic. For example the first order (FO) theory of the naturals with addition and multiplication (the elementary arithmetic) is algorithmically undecidable (see [1]). When a “full” theory is undecidable then its special fragments are investigated (as in [2]). For example if we “remove” multiplication from the elementary arithmetic then the obtained theory became decidable [1]. However, the theoretical decidability of a theory may be useless if any decision algorithm needs extremely high resources (time and space). So the computational complexity of decidable theories is investigated. Some results of the computational complexity theory can be found in [3].

One of central results in the computational complexity theory is obtained by Fischer and Rabin in [4]. This theorem connects the definability of arithmetic operations in the theory and the computational complexity of this theory. The last depends on length of defining formulas. In [4] it was proved that the additive FO-arithmetic of the reals needs at least $2^{\Theta(n)}$ time for formulas of length $n$, and the additive FO-arithmetic of the naturals needs at least $2^{2^{\Theta(n)}}$ time to be decided.

Due to the incompleteness for PTIME (see [5]) the language of FO-logic is often extended by some non-FO features. One of them is the transitive closure (TC) operator. The TC-operator is not FO-definable in the general case (see [5]) so this operator can increase the expressive power of theories and their computational complexity. The importance of the TC-operator for abstract computational complexity theory was established by Immerman in [6, 7]. The expressive power and the computational complexity of TC-logic for finite structures are reviewed in [8]. These investigations has a value for database theory because logic languages are widely used as query languages in databases and other datastore systems (see [9]).

Here we investigate the TC-operator over an infinite structure with the successor function. This function and its theories are intensively studied in many works (see for example [10]).
The FO-theory of the successor function is a PSPACE-complete problem (see [11]). Hence this theory is decidable in polynomial space and exponential time. Non-FO features can increase the complexity, for example, the next classical result was established by Meyer in [12]. The weak monadic second-order theory (i.e., a theory allowing second order quantification over finite subsets only) of the successor function (this theory is named WS1S) needs hyperexponential time and space to be decided.

The theory of the successor function is undecidable when the $k$-ary TC-operators are allowed for some $k > 1$. So we investigate the successor function theory with the unary TC-operator only. The discrete linear order relation is definable via the successor function and the unary TC-operator. In the other way the successor function is FO-definable via the discrete linear order. Hence we have a structure with the successor function and the discrete linear order and we consider the theory of this structure with the unary TC-operator (we denote such a theory by TDO).

There exist discrete partial orders with undecidable FO-theories (see [13] for example), but the decidability of the discrete linear order FO-theory is well known. The last theory is also a PSPACE-complete problem (see [11]). But with the unary TC-operator and the successor function we can express more than only the discrete linear order. For example the unary relations “divisible by $c$” are definable for all natural $c$. Thus, the computational complexity of TDO can be higher than PSPACE.

The unary TC-operator is definable in WS1S, so the decidability of TDO is followed from [12]. Also Meyer’s result implies the hyperexponential upper bound for the computational complexity of TDO. An explicit algorithm of such complexity was constructed in [14]. But the second order quantifiers can’t be expressed in TC-logic so a lower bound of the TDO complexity can’t be obtained the same way.

We use the Fischer-Rabin theorem to establish this lower bound. If we extend TDO by one of arithmetic operations (addition or multiplication) then the obtained theory became undecidable (see [15]). So both arithmetic operations can be expressed in TDO in limited intervals only. In our previous paper [16] we have found formulas for these representations. Now we use these results to obtain the lower bound for the computational complexity of TDO.

Our main result is the following: for any decision algorithm $A$ for TDO there are infinitely many formulas $\phi$ such that $A$ works on $\phi$ at least $\Upsilon_2(\Theta(n))$ time and needs at least $\Upsilon_2(\Theta(n))$ space. Here $n$ is length of $\phi$, $\Upsilon_2$ is the hyperexponential function:

$$\Upsilon_2(n) = \Upsilon_2^{n \times 2, 2}, \quad \Upsilon_2(0) = 1.$$  

Using the previous results we obtain the completeness of TDO in the corresponding complexity class. Therefore, we prove that the unary TC-operator drastically increases the computational complexity of the theories of the discrete linear order and the successor function.

2. Preliminaries and Definitions

We denote length of a word $w$ by $|w|$. The usual Landau notation $O(x)$ means a value with an asymptotic upper bound of the form $cx$ where $c$ is a positive constant: $O(x) \leq cx$. The notation $\Theta(x)$ means a value $y$ such that $y = O(x)$ and $x = O(y)$, i.e. the inequality $c_1 x \leq \Theta(x) \leq c_2 x$ holds asymptotically for some positive constants $c_1$ and $c_2$.

In [4] the following result is proved (theorems 6 and 7). This is a main tool to establish lower bounds of the computational complexity for arithmetic theories.

**Theorem 1** (Fischer-Rabin Theorem). *Let $T$ be a theory of the (natural, real) numbers with addition, $f$ be a function growing exponentially or faster. Let there exist a polynomial-time algorithm that by a word $w$ of length $n$ constructs following formulas of length $O(n)$:
Theorem 3. \( I_n(x) \) meaning in \( T \) that \( x < f(n)^2 \); 
\( J_n(x) \) meaning in \( T \) that \( x = f(n) \); 
\( S_n(x,y) \) coding in \( T \) binary sequences \( \alpha \) of length \( f(n)^2 \). I. e. for any binary sequence \( \alpha \) of length \( f(n)^2 \) there exists a “code” (number) \( a_\alpha \) such that for all \( i = 0, \ldots, f(n)^2 - 1 \) the formula \( S_n(a_\alpha, i) \) is true in \( T \) if and only if \( \alpha_i = 1 \), where \( \alpha_i \) is the \( i \)-th element of \( \alpha \); 
\( H_\omega(x) \) meaning in \( T \) that \( x \) is a “code” of the sequence \( w^0 \), \( p = f(n) - n \).

Then for any decision algorithm \( A \) for \( T \) there are infinitely many formulas \( \phi \) such that \( A \) works on \( \phi \) at least \( f(\Theta(|\phi|)) \) time.

Really it has been proved that such a theory \( T \) is a hard problem in the complexity classes \( \text{TIME}(f(\Theta(n))) \) and \( \text{NTIME}(f(\Theta(n))) \), i. e. all problems from corresponding classes are polynomial-time reducible to \( T \).

Note that in the proof of the Fischer-Rabin Theorem all second summands are bounded: terms of kind \( x + y \) are used only for \( y \leq \Delta \) where \( \Delta = O(f(n)) \). Total number of additions in the constructed formulas are limited. Hence this theorem holds if there exist some polynomial-time constructible formulas \( \text{sum}_\Delta(x, y, z) \) of length \( O(n) \) describing “bounded” addition:

\[
\text{sum}_\Delta(x, y, z) \equiv [x + y = z \land y \leq \Delta].
\]

In this paper we consider formulas containing as first-order language features (boolean connectors, quantifiers, see [1]) so the unary transitive closure (TC) operator:

**Definition 1** (see [8]). Let \( \psi(x, y, z) \) be a formula with free variables \( x, y, z \) (\( z \) is a tuple). Then \( T_{x,y} \psi \) is a formula with free variables \( x, y, z \).

The formula \( (T_{x,y} \psi)(x, y, z) \) is true when there exists a finite sequence of universe elements \( a_0 = x, a_1, \ldots, a_n = y \) and the formulas \( \psi(a_i, a_{i+1}, z) \) are true for all \( i = 0, \ldots, n - 1 \).

We consider the successor function \( s \) (i. e. a one-to-one acyclic unary function) defined over any infinite universe. Using this function \( s \) and the unary TC-operator we can define the discrete linear order without last element:

\[
x \leq y \equiv T_{x,y}(y = s(x)).
\]

And vice versa the function \( s \) is definable via the order relation: \( s(x) \) is the smallest of elements greater than \( x \). So we consider the discrete linear ordered universe without last element and the theory of such a structure with the unary TC-operator. Let us denote this theory by TDO.

The presence of a first element has no value, we can select anyone as first. Further we denote it by 0. We never use “negatives” (elements less than 0) so we can suppose that the considered structure is the usually ordered natural numbers. The strict \(<\) and non-strict \(\leq\) order relations are definable one by the other hence we can use both.

In [16] we have proved the next two claims for TDO.

**Theorem 2.** For any formula \( \text{sum}_\Delta(x, y, z) \) there is a formula \( \text{sum}_\delta(x, y, z) \) where \( \delta = \text{LCM}(1, \ldots, \Delta) \), \( |\text{sum}_\delta| \leq |\text{sum}_\Delta| + \text{const} \). This formula \( \text{sum}_\delta \) is polynomial-time constructible by \( \text{sum}_\Delta \).

Here \( \text{LCM}(1, \ldots, \Delta) \) is the least common multiple of \( \{1, 2, 3, \ldots, \Delta\} \).

**Theorem 3.** For any formula \( \text{sum}_\Delta(x, y, z) \) of length \( m \) there are the following formulas of length \( O(m) \):

- \( \text{const}_\Delta(x) \) meaning that \( x = \Delta \);
- \( \text{div}_\Delta(x, y) \) meaning that \( y \) is divisible by \( x \) and \( x \leq \Delta \);
• \(\text{sqr}_\Delta(x, y)\) meaning that \(x^2 = y\) and \(x, y \leq \Delta\);
• \(\text{pow}_\Delta(x, y)\) meaning that \(2^x = y\) and \(x, y \leq \Delta\).

These formulas \(\text{const}_\Delta, \text{div}_\Delta, \text{sqr}_\Delta, \text{pow}_\Delta\) are polynomial-time constructible by \(\text{sum}_\Delta\).

A general idea of these theorems proofs is the following. By the formula \(\text{sum}_\Delta\) we define \(\text{div}_\Delta\) with the unary TC-operator:

\[
\text{div}_\Delta(x, y) \equiv (T_{x, y}\text{sum}_\Delta(z, x, y))(0, x, y),
\]
then by \(\text{div}_\Delta\) we define \(\delta = \text{LCM}(1, \ldots, \Delta).\) The formula \(\text{sum}_\delta\) is constructed by addition modulo \(u\) for \(u \leq \Delta.\) The formulas \(\text{sqr}_\Delta\) and \(\text{pow}_\Delta\) are defined by \(\text{div}_\Delta\).

3. Main Results

Our main result is the following theorem.

**Theorem 4.** TDO is a hard problem in the class \(\text{TIME}(\Upsilon_2(\Theta(n)))\). Therefore for any decision algorithm \(A\) for TDO there are infinitely many formulas \(\phi\) such that the algorithm \(A\) works on \(\phi\) at least \(\Upsilon_2(\Theta(|\phi|))\) time.

**Proof.** Using the second Chebyshev function \(\Psi:\)

\[
\Psi(x) = \sum_{p^m \leq x} \ln p
\]
and de la Vallée-Poussin’s asymptotic approximation (see [17]):

\[
\Psi(x) = x + O(xe^{-\Theta(\sqrt{\ln x})})
\]
it is easy to obtain the asymptotic inequality:

\[
\text{LCM}(1, \ldots, x) = e^{\Psi(x)} > 2^x.
\]

Indeed, we have \(\Psi(x) = x + O(x\epsilon)\) where \(\epsilon\) tends to 0 as \(x \to \infty\), hence the inequality \(\Psi(x) > x \ln 2\) holds asymptotically since \(\ln 2 < 1\). Let us select a natural \(\Delta_0\) such that these inequalities holds for all \(x \geq \Delta_0\). It is obvious that \(\Delta_0 \geq 4\) since \(\text{LCM}(1, 2, 3) = 6 < 2^3\).

We define a function \(f\) as

\[
f(0) = \Delta_0;
\]
\[
f(n + 1) = \text{LCM}(1, \ldots, f(n)).
\]
From the previous we obtain that \(f(n+1) > 2^{f(n)}\). Hence for any natural \(n\) we get \(f(n) > \Upsilon_2(n)\), it can be easy established by induction on \(n:\)

\[
f(0) = \Delta_0 \geq 4 > 1 = \Upsilon_2(0);
\]
\[
f(n + 1) > 2^{f(n)} > 2^{\Upsilon_2(n)} = \Upsilon_2(n + 1).
\]
Note that the inequality \(x^2 \leq 2^x\) holds for all \(x \geq 4\). The last implies that \(f(n + 1) \geq (f(n))^2\) for all \(n\) because \(f(n) \geq \Delta_0 \geq 4\).

By Theorem 2 for any natural \(n\) we can construct the formula \(\text{sum}_{f(n)}(x, y, z)\) of length \(O(n)\). These formulas are constructed by induction on \(n:\) in the formula \(\text{sum}_{f(0)}\) we apply the successor function \(s y \leq \Delta_0\) times to \(z\), and then we use Theorem 2 \(n\) times.

As there is the formula \(\text{sum}_{f(n)}(x, y, z)\) of length \(O(n)\) so by Theorem 3 there is the formula \(\text{const}_{f(n)}(x)\) of length \(O(n)\).
Generally our method likes one from [4].

Theorem. All these formulas are of length $f(n)$.

As there is the formula $\sum_{f(n+1)}(x, y, z)$ of length $O(n)$ so by Theorem 3 there is the formula $\text{sqr}_{f(n+1)}(x, y)$ of length $O(n)$. Let us note that

$$f(n + 1) \geq 2f(n) \geq (f(n))^2.$$

As there is the formula $\sum_{f(n+2)}(x, y, z)$ of length $O(n)$ so by Theorem 3 there is the formula $\text{pow}_{f(n+2)}(x, y)$ of length $O(n)$. Note that

$$f(n + 2) \geq 2f(n+1) \geq 2(f(n))^2.$$

Now we can construct formulas $I_n$, $J_n$, $S_n$, $H_n$ needed for the Fischer-Rabin Theorem. Generally our method likes one from [4].

The formulas $I_n$ and $J_n$ are

$$J_n(x) \equiv \text{const}_{f(n)}(x);$$

$$I_n(x) \equiv (\exists u)(\exists v)(\text{const}_{f(n)}(u) \land \text{sqr}_{f(n+1)}(u, v) \land x < v).$$

The formula $S_n$ must “code” binary sequences of length $(f(n))^2$. We use the binary numeral system. Informally given any binary sequence $\alpha$ we “reverse” it from right to left and the “reversed” sequence is the binary representation of some natural $a_\alpha$. This $a_\alpha$ is our “code” for $\alpha$.

Remember that $2^n - 1$ is the greatest natural with the binary representation of length $n$. So for binary sequences of length $(f(n))^2$ we need to operate with numbers up to $2(f(n))^2$. The last is less than $2^{2f(n)}$ and less than $f(n+2)$. Hence the formula $S_n$ can be constructed as

$$S_n(x, y) \equiv (\exists t)(\exists u)(\exists v)(\exists w)(\exists z)(\text{pow}_{f(n+2)}(y, u) \land \sum_{f(n+2)}(u, u, t) \land v < u \land \text{div}_{f(n+2)}(t, w) \land \sum_{f(n+2)}(v, u, z) \land \sum_{f(n+2)}(z, w, x)).$$

Here we write that “1” is in the binary number $x$ on the $y$-th significant position. The variable $u$ contains the $y$-th position, $v$ contains the lower positions, and $w$ contains the higher ones.

The formula $H_w$ must define the number $x_w$ for that $w$ is the “reversed” binary representation. First we construct an additional formula $H'_w$. The formula $H'_w$ defines $x_w$ yet, but $H'_w$ is longer than $O(|w|)$. We use induction by $|w|$ to construct $H'_w$. In the empty word $\Lambda$ there is no “1” so

$$H'_\Lambda(x) \equiv [x = 0].$$

For non-empty words $w$ we define $H'_w$ by the suffix of $w$:

$$H'_{0w}(x) \equiv (\exists u)(H'_w(u) \land \sum_{f(n)}(u, u, x));$$

$$H'_{1w}(x) \equiv (\exists u)(H'_w(u) \land \sum_{f(n)}(s(u), u, x)).$$

If $|w| = n$ then the formula $H'_w$ is of length $O(n^2)$ due to the conjunction of $n$ formulas of kind $\sum_{f(n)}(a, b, c)$. These formulas are of length $O(n)$. But this conjunction can be replaced with the shorter formula using the equivalence

$$\bigwedge_{i=1}^{n} \sum_{f(n)}(a_i, b_i, c_i) \equiv (\forall a)(\forall b)(\forall c) \left[ \bigvee_{i=1}^{n} (a = a_i \land b = b_i \land c = c_i) \right] \rightarrow \sum_{f(n)}(a, b, c).$$

In this equivalence both parts of the right implication are of length $O(n)$. The formula $H_w$ is obtained from $H'_w$ with such a substitution.

Thus, it is proved that we can construct in polynomial time the formula $\sum_{f(n)}$ defining addition up to $f(n)$ in TDO, and the formulas $I_n$, $J_n$, $S_n$ and $H_w$ needed for the Fischer-Rabin Theorem. All these formulas are of length $O(n)$.

By the Fischer-Rabin theorem TDO must be a $\text{TIME}(f(\Theta(n)))$-hard problem, and $\text{TIME}(\Upsilon_2(\Theta(n)))$-hard consequently.
Corollary 1. TDO is a SPACE(\(\Upsilon_2(\Theta(n))\))-hard problem, hence, for any decision algorithm \(A\) for TDO there are infinitely many formulas \(\phi\) such that the algorithm \(A\) on \(\phi\) needs at least \(\Upsilon_2(\Theta(|\phi|))\) space.

Proof. Well known (see [3]) that time \(T\) and space \(S\) of a computation satisfy the inequality \(T \leq 2^O(S)\), so \(S \geq \Omega(\log_2 T)\). If \(T \geq \Upsilon_2(\Theta(n))\) then

\[
S \geq \Theta(\log_2 \Upsilon_2(\Theta(n))) = \Theta(\Upsilon_2(\Theta(n) - 1)) = \Upsilon_2(\Theta(n)).
\]

Combining these claims and results from [12, 14] we get

Corollary 2. TDO is complete in the complexity class \(\text{TIME}(\Upsilon_2(\Theta(n))) = \text{SPACE}(\Upsilon_2(\Theta(n)))\).

4. Conclusion
We have proved that the unary transitive closure (TC) operator increase the computational complexity of the discrete linear order theory and the successor function theory. Without the TC-operator these theories are polynomial-space (and exponential-time) decidable. The TC-operator increases the computational complexity to \(\Upsilon_2(\Theta(n))\) time and space.

This result has as the theoretical so the practical value. Iterative operators (TC is one of them) are widely used in modern query languages for database systems (see [8]). Hence the lower bound \(\Upsilon_2(\Theta(n))\) implies that queries written in such languages can be very hard computable.

We are interested in the following questions. What about other theories with and without the TC-operator? Is there non-trivial example for TC-theory with tractable computational complexity? And what about other non-FO theories for the successor function or the discrete linear order?

References
[1] Boolos G S, Burgess J P and Jeffrey R C 2007 Computability and Logic 5th ed (Cambridge: Cambridge University Press)
[2] Lechner A, Ouaknine J and Worrell J 2015 LICS (IEEE Computer Society) pp 667–676
[3] Arora S and Barak B 2009 Computational Complexity: A Modern Approach 1st Edition 1st ed (Cambridge: Cambridge University Press)
[4] Fischer M J and Rabin M O 1974 Proceedings of the SIAM-AMS Symposium in Applied Mathematics vol 7 pp 27–41
[5] Aho A V and Ullman J D 1979 POPL (ACM Press) pp 110–120
[6] Immerman N 1987 SIAM J. Comput. 16 760–778
[7] Immerman N 1988 SIAM J. Comput. 17 935–938
[8] Libkin L 2013 Elements of Finite Model Theory (Berlin Heidelberg: Springer-Verlag)
[9] Surinx D, Fletcher G H L, Gyssens M, Leinders D, Van den Bussche J, Van Gucht D, Vansummeren S and Wu Y 2015 Logic Journal of the IGPL 23 759–788
[10] Michel P 2007 Arch. Math. Log. 46 123–148
[11] Wagner K and Wechsung G 1986 Computational Complexity (Dordrecht: D. Reidel Publishing Company)
[12] Meyer A R 1975 Logic Colloquium ed Parikh R (Berlin, Heidelberg: Springer Berlin Heidelberg) pp 132–154 ISBN 978-3-540-37483-1
[13] Halfon S, Schnoebelen P and Zetzsche G 2017 2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS) pp 1–12
[14] Zolotov A S 2015 Lobachevskii Journal of Mathematics 36 434–440 ISSN 1818-9962
[15] Zolotov A S 2014 Vestnik TvGU. Seriya Prikladnaya matematika [Herald of Tver State University. Series: Applied mathematics] 3 117–127 (in Russian)
[16] Dudakov S M 2017 Vestnik TvGU. Seriya Prikladnaya matematika [Herald of Tver State University. Series: Applied mathematics] 4 7–15 (in Russian)
[17] Karatsuba A A 1992 Basic Analytic Number Theory (Berlin Heidelberg: Springer-Verlag)