# Distributional Word Problem for Tseitin Semigroup

Arist Kojevnikov[1], Ivan Monakhov[2], Sergey K. Naumov[2]

[1] St.Petersburg Department of V. A. Steklov Institute of Mathematics
St.Petersburg, Russia
http://logic.pdmi.ras.ru/~arist/
[2] St.Petersburg State University, St.Petersburg, Russia

**Abstract.** The main criticism of known algebraic distributional NP (DistNP) complete problems is based on the fact that they contain too many specific relations to simulate a Turing machine. In this paper we present a construction of the semigroup with very few relations and word problem that is DistNP complete. Our construction follows Tseitin ideas [Tse56]. We modify original construction to work with words in standard binary presentation and arbitrary semigroups without any special conditions on its relations.

The study of average case complexity (i.e. complexity of algorithms for problems with probability distribution on instances) is hard and interesting from many points of view. For example, in industry, it is interesting to understand the behavior of programs on most common inputs, or in cryptography, hardness of a cipher in the worst case is not too interesting.

In [Lev86] Levin defined a notion of *distributional NP-complete problem* (DistNP-complete) where decision problem component belongs to NP and every distributional problem consisting of an NP problem and a polynomial-time computable distribution is reducible to it. The quest for usable and natural DistNP-complete problem is not finished, although a lot of work were done in this direction. Matrix transformation [BG95], matrix representability [VR92], word problem for semigroups [WB95] and word problem for groups [Wan99] have been shown to be DistNP-complete. From another side many important questions are still open. For example, no construction of the simplest cryptographic primitive such as one-way function from DistNP-complete problem is known [Lev03]. For further information on average case complexity we refer the reader to surveys [Gur91,Wan97,BT06].

In this work we propose a new simple construction of DistNP-complete problem that is based on the one of the most compact (in the overall representation) undecidable algebraic problem, the word problem for Tseitin semigroup [Tse56]. The main idea of original construction is the following: given a semigroup $G$ with alphabet of 2 symbols $a$ and $b$ (where $a$ codes a separator), with hard word problem, we code all relations in $G$ as a prefix in the alphabet of $c, d$ (where $c$ codes $a$ and $d$ codes $b$) and use an additional symbol $e$ to extract relations from the prefix to simulate derivation in $G$.

To fit the word problem for Tseitin semigroup in the framework of average-case complexity we have made two modifications in the construction. The first one is that we work with semigroup $G$ where $a$ is not the separator. To present symbol $a$ in the prefix we use word $d$ and to present $b$ we use $dd$. Symbol $c$ works as separator as before. The second modification is the following: in the original construction semigroup $G$ has in all relations an empty word at the right side. Only one known construction of such semigroup follows from the group with a hard word problem, i.e. Boone-Novikov construction. It is hard in it to check the equivalence of two very special words: $(xkx^{-1})t$ and $t(xkx^{-1})$. Since this word problem is not natural at all, we modify the Tseitin construction such that it works with any semigroup in "standard" representation: $< a, b \mid K_t \leftrightarrow L_t, t = 1..m >$, where $K_t, L_t$ are not empty.

All known DistNP-complete problems have two parts: one is random word and the second one is encoding of random Turing Machine. The probability of random instance to code some Turing Machine is a constant, but very small constant [Mya07]. In our construction any prefix in $\{c, d\}^*$ that does not contain word $ddd$ and use $c^k, k \geq 3$ as separator between rules codes some Turing Machine. So if we extend the semigroup with relation $ddd = dd$, then any random prefix will code a Turing Machine. We can also use strings in binary alphabet with uniform distribution as inputs for our problem: if we get a string $X$, then transform first $O(\log |X|)$ bits to the encoding $S$ of some semigroup (using substitution $\{0 \leftarrow c, 1 \leftarrow d\}$ and writing constant number of $ccc$) and additional words and use other symbols in $X$ as $x$.

The paper is organized as follows. Sect. 1 contains the necessary definitions. The proof of the main result follows [Wan99,AD00] and is presented in Sect. 2.

## 1 Preliminaries

In this section we recall basic definitions and results of average-case complexity that we will use.

Let $\Sigma$ denote some alphabet. A real-valued function $\mu : \Sigma^* \to [0, 1]$ is *probability distribution* (or just *distribution*) iff $\sum_{x \in \Sigma^*} \mu(x) = 1$. We assume that $\mu(\Lambda) = 0$, where $\Lambda$ is an empty string. We call the pair $(A, \mu)$ a *distributional problem*, where $A$ is a decision problem and $\mu$ is a distribution over inputs for $A$. For distributions $\mu$ and $\nu$, $\mu$ is *dominated* by $\nu$ ($\mu \preceq \nu$), if there is a polynomial $p$ such that, for all $x$, $\mu(x) \leq p(|x|)\nu(x)$.

**Definition 1.** *We say that distributional problem $(A, \mu)$ is* polynomial time reducible *to distributional problem $(B, \nu)$ if there is a polynomial time computable function $f$ (we call it* reduction*) such that $x \in A \Leftrightarrow f(x) \in B$ and $\mu$ is* dominated by $\nu$ with respect to $f$*, i.e. there is a distribution $\mu_1$ on $A$ such that $\mu \preceq \mu_1$ and $\nu(y) = \sum_{f(x)=y} \mu_1(x)$.*

$(A, \mu)$ is in DistNP if $A$ is in NP, and $\mu$ is polynomial-time computable.

If a semigroup $G$ has alphabet $a_1, \ldots, a_n$ and relation set $K_1 \leftrightarrow L_1, \ldots, K_m \leftrightarrow L_m$ we will write $G =< a_1, \ldots, a_n \mid K_1 \leftrightarrow L_1, \ldots, K_m \leftrightarrow L_m >$. We will also

write $A \leftrightarrow B$ if words $A$, $B$ are equivalent in $G$. If moreover $A = UK_iV$ and $B = UL_iV$ for some $i = 1..m$ we will call $A \leftrightarrow B$ *elementary transformation.*

To prove our completeness result we will use the following important lemma formulated in [WB95].

**Lemma 1 (Distribution controlling lemma).** *Let $\mu$ be a polynomial-time computable distribution. Then there is a total polynomial-time computable and polynomial-time invertible one to one function $\alpha : \Sigma^* \to \Sigma^*$ such that for all $x$, $\mu(x) < 4 \cdot 2^{-|\alpha(x)|}$.*

*Also, if there is a polynomial $p$ such that for all $x$, $\mu(x) > 2^{-p(|x|)}$, then there is a total polynomial-time computable and polynomial-time invertible one to one function $\beta : \Sigma^* \to \Sigma^*$, such that for all $x$ $4 \cdot 2^{-|\beta(x)|} \leq \mu(x) < 20 \cdot 2^{-|\beta(x)|}$.*

## 2 Word Problem for Modified Tseitin semigroup

**Definition 2.** *The* distributional word problem for Tseitin semigroup *is the following decision problem:*

*Given a semigroup*

$$Ts = < a, b, c, d, e \mid ac = ca, ad = da, bc = cb, db = bd, ccc = eccc,$$
$$ccce = ccc, acdec = ecdc, cdce = cedca, bcddec = ecddc, cddce = ceddcb >$$

*where $x, h \in \{a, b\}^*$, $S \in \{c, d\}^*$ and a natural number $n$ is in unary notation. Is it possible to obtain $Sh$ from $Sx$ in $Ts$ in at most $n$ elementary transformations?*

*The distribution is the following: randomly uniformly and independently select number $n$ and strings $x, h \in \{a, b\}^*$, $S \in \{c, d\}^*$. It is proportional to*

$$\frac{2^{-(|x|+|S|+|h|)}}{(n|x||h||S|)^2}.$$

**Theorem 1.** *The distributional word problem for Tseitin semigroup is DistNP-complete.*

*Proof.* We follow the proof of undecidability of Tseitin semigroup presented in [Tse56,AD00]. The main difference is that we use another definition of prefix presenting a semigroup with hard word problem.

Let $(D, \mu)$ be a distributional problem in DistNP. From distribution controlling lemma, there is a total polynomial-time computable and polynomial-time invertible one to one function $\alpha : \{0,1\}^* \to \{0,1\}^*$ such that $\mu(x) \preceq 2^{-|\alpha(x)|}$.

**Lemma 2 ([WB95,Wan99]).** *There is a semigroup $G$ with alphabet $\{0, 1\}$ and $|G| = O(\log(|x|))$ such that $x \in D \iff \underline{s}1\alpha(x)\underline{\$} \leftrightarrow \underline{h}$ in $G \iff \underline{s}1\alpha(x)\underline{\$} \leftrightarrow \underline{h}$ using polynomial in $|x|$ number of elementary transformations in $G$, where $\underline{s}$, $\underline{\$}$ and $\underline{h}$ are words in $G$ with length $2\log(x) + O(1)$.*

Let a semigroup $G$ be the semigroup from Lemma 2 and the following finite presentation:

$$< a, b \mid K_j \leftrightarrow L_j,\ j = 1..m >,\ K_j, L_j \neq \Lambda.$$

**Definition 3.** *We define the mapping $\tau$ by induction on length of $X$ in $G$:*

$$\tau(\Lambda) = \Lambda, \tau(Xa) = \tau(X)cd, \tau(Xb) = \tau(X)cdd.$$

*For nonempty word $X$ we will use the notation $(X]$ to denote a word obtained from $X$ by removing the first symbol of $X$. Note, that for any nonempty word $X \in \{a, b\}^*$ it is true that*

$$\tau(X) = c(\tau(X)].$$

We will use a word $S$ to code all relations in $G$:

$$S = cc\tau(K_1)c\tau(L_1)cc\tau(K_2)c\tau(L_2)\dots c\tau(L_m)ccc.$$

Also we need the following notation for reversing string in alphabet $\{c, d\}$:

$$\overline{\Lambda} = \Lambda, \overline{Xc} = c\overline{X}, \overline{Xd} = d\overline{X}.$$

**Lemma 3.** *For any words $P, Q \in \{a, b\}^*$*

$$P \leftrightarrow Q \text{ in } G \Longleftrightarrow SP \leftrightarrow SQ \text{ in } Ts,$$

*moreover number of elementary transformations in both derivations are polynomially dependent.*

*Proof.* $\Longrightarrow$: Let

$$P = UL_tV, Q = UK_tV.$$

It is easy to check that:

$$\begin{aligned}
&SQ = S_1cc\tau(K_t)c\tau(L_t)cccS_2UL_tV \leftrightarrow S_1Ucc\tau(K_t)cL_t\tau(L_t)cccS_2V\\
&\leftrightarrow S_1Ucc\tau(K_t)cL_t\tau(L_t)ecccS_2V \leftrightarrow S_1Uccc(\tau(K_t)]ce\tau(L_t)cccS_2V\\
&\leftrightarrow S_1Ucccе(\tau(K_t)]cK_t\tau(L_t)cccS_2V \leftrightarrow S_1Ucc\tau(K_t)cK_t\tau(L_t)cccS_2V\\
&\leftrightarrow S_1cc\tau(K_t)c\tau(L_t)cccS_2UK_tV = SP
\end{aligned}$$

To prove the reverse direction we need the following functions:

1. projections $P_{a,b}(X)$ and $P_{c,d}(X)$ of word $X$ on alphabets $\{a, b\}$ and $\{c, d\}$ respectively, i.e. take a word $X$ and remove all symbols different from $a, b$ and $c, d$ respectively;
2. function $\tau'$, almost reverse to $\tau$:

$$\tau'(X) = \begin{cases} X, & \text{iff } P_{c,d}(X) = Ucccc,\\ \tau'(P_{a,b}(X)Uc)\overline{\tau(\alpha)}\alpha, & \text{iff } P_{c,d}(X) = Uc\overline{\tau(\alpha)},\\ \tau'(VU)\tau(\alpha), & \text{iff } P_{c,d}(X) = U\tau(\alpha) \text{ and } P_{a,b}(X) = V\alpha,\\ \text{is not defined,} & \end{cases}$$

   where $\alpha \in \{a, b\}$.

3. extractor $\xi$ of the word in $G$ from the word in $Ts$: $\xi(U) = P_{a,b}(U)$,

$$\xi(UeV) = \begin{cases} \xi(\tau'(U)V), \text{ iff } P_{c,d}(U) = S_1 ccc\tau(K)c \text{ or} \\ \qquad\qquad P_{c,d}(U) = S_1 ccc\tau(K)c\tau(L) \text{ and } P_{a,b}(U) = U_1 L \\ \xi(U\overline{\tau'(\overline{V_1 ccc})}W), \text{otherwise,} \end{cases}$$

where $V = V_1 cccW$ is a word in $Ts$, $U, V_1$ are words in $Ts$ without $e$ and $P_{c,d}(UV) = S$. Informally, we take a word in $Ts$ and remove all $e$ by moving them to $ccc$ in one virtual step, after that we project the word without $e$ to $\{a, b\}^*$.

To finish the proof of the lemma we need the following fact:

**Lemma 4.** *For words $P \in \{a, b\}^*$ and $X \leftrightarrow SP$ in $Ts$ the following holds in semigroup $G$:*

$$\xi(X) \leftrightarrow P.$$

*Proof.* Let $P$ be a word in the alphabet $\{a, b\}$ and assume that $X$ can be obtained by the following sequence of elemental transformations in $Ts$:

$$SP = X_0 \leftrightarrow X_1 \leftrightarrow \ldots \leftrightarrow X_k \leftrightarrow X_{k+1} = X.$$

It is easy to see that for all $X_j$, $j = 0..k+1$ holds $P_{c,d}(X_j) = S$.

We prove by induction on $j$. The base is true since $\xi(SP) = P$. Assume that the lemma statement is proved for $X_k$. We need to show that $\xi(X_k) = \xi(X_{k+1})$. Consider the following cases:

1. $X_j = U\alpha\beta V$, $X_{j+1} = U\beta\alpha V$, for $\alpha \in \{a, b\}$, $\quad \beta \in \{c, d\}$:
   If $U = U_1 cccU_2$ and $V = V_1 cccV_2$, where $U_2$ and $V_1$ do not contain $e$, then we can remove all $e$ in $U_1, V_2$ and

   $$\xi(X_j) = P_{a,b}(U\alpha\beta V) = P_{a,b}(U\beta\alpha V) = \xi(X_{j+1}).$$

   Otherwise after a number of applications of $\tau'$ to both sides, the difference between $\xi(X_j)$ and $\xi(X_{j+1})$ will disappear. Function $\tau'$ is independent to an application of the current rule, i.e. $\tau'(U\alpha\beta V) = \tau'(U\beta\alpha V)$.
2. $X_j = U\alpha\tau(\alpha)ecV$, $X_{j+1} = Ue\tau(\alpha)cV$, for $\alpha \in \{a, b\}$:
   Consider the case when $P_{c,d}(U) = U_1 ccc\tau(K)c\tau(L)$, $P_{a,b}(U) \neq U_1 L$ and $V = V_1 cccW$:

   $$\begin{aligned} &\xi(X_j) = \xi(U\alpha\tau(\alpha)ecV_1 cccW) \\ &= \xi(U\alpha\tau(\alpha)\overline{(\tau'(ccc\overline{V}_1 c))}W) = \xi(U\overline{(\tau'(ccc\overline{V}_1 c)\overline{\tau(\alpha)}\alpha)}W_1) \\ &= \xi(U\overline{(\tau'(ccc\overline{V}_1 c\overline{\tau(\alpha)}))}W) = \xi(Ue\tau(\alpha)cV_1 cccW) = \xi(X_{j+1}), \end{aligned}$$

   otherwise:

   $$\begin{aligned} &\xi(X_j) = \xi(U\alpha\tau(\alpha)ecV) = \xi(\tau'(U\alpha\tau(\alpha))cV) \\ &= \xi(\tau'(Uc)\tau(\alpha)cV) = \xi(Uce\tau(\alpha)cV) = \xi(X_{j+1}). \end{aligned}$$

3. $X_j = Uc\overline{\tau(\alpha)}eV$, $X_{j+1} = Uce\overline{\tau(\alpha)}\alpha V$, for $\alpha \in \{a, b\}$:
in this case $P_{c,d}(U) = U_1 ccc\tau(K)c$:

$$\xi(X_j) = \xi(Uc\overline{\tau(\alpha)}eV) = \xi(\tau'(Uc\overline{\tau(\alpha)})V)$$
$$= \xi(\tau'(Uc)\overline{\tau(\alpha)}\alpha V) = \xi(Uce\overline{\tau(\alpha)}\alpha V) = \xi(X_{j+1}).$$

4. $X_j = UccceV$, $X_{j+1} = UcccV$:

$$\xi(X_j) = \xi(UccceV) = \xi(\tau'(Uccc)V) = \xi(UcccV) = \xi(X_{j+1}).$$

5. $X_j = UecccV$, $X_{j+1} = UcccV$:
Consider the case when $P_{c,d}(U) = U_1 ccc\tau(K)c\tau(L)$, $P_{a,b}(U) \neq U_1 L$:

$$\xi(X_j) = \xi(UecccV) = \xi(U\overline{\tau'(\overline{ccc})}V) = \xi(U\overline{ccc}V) = \xi(UcccV) = \xi(X_{j+1}),$$

otherwise (in the case with $P_{a,b}(U) = ZcccU_2L$, $U_2$ is a word without $ccc$ and $e$) note that $G$ contains rule $K \leftrightarrow L$:

$$\xi(X_j) = \xi(UecccV) = \xi(\tau'(U)cccV) = \xi(ZcccU_2KP_{c,d}(U)U_2cccV)$$
$$\leftrightarrow \xi(ZcccU_2LP_{c,d}(U)U_2cccV) = \xi(\tau'(U)cccV) = \xi(UcccV)\xi(X_{j+1}).$$

□

$\Longleftarrow$: Assume that $SP \leftrightarrow SQ$ in $Ts$ and use Lemma 4:

$$P = \xi(SP) \leftrightarrow \xi(SQ) = Q.$$

□

From Lemma 2 and Lemma 3 we have the following:

**Corollary 1.** *There is a word $S \in \{c, d\}^*$ with length at most $O(\log(n))$ such that $x \in D \iff S\underline{s}b\underline{\alpha(x)}\underline{\$} \leftrightarrow S\underline{h}$ in $Ts \iff S\underline{s}b\underline{\alpha(x)}\underline{\$} \leftrightarrow S\underline{h}$ using polynomial in $|x|$ number of elementary transformations in $Ts$, where $\underline{s}$, $\underline{\$}$ and $\underline{h}$ are words in $\{a, b\}$ with length $2\log(x) + O(1)$, string $\underline{\alpha(x)}$ is obtained from $\alpha(x)$ by substitution $\{0 \leftarrow a, 1 \leftarrow b\}$.*

Now we are ready to finish the proof of Theorem 1. We define a reduction $f$ as follows:

$$f(x) = (S, \underline{s}b\underline{\alpha(x)}\underline{\$}, \underline{h}).$$

The probability distribution of $f(x)$, that is proportional to

$$\frac{2^{-(|S|+|\underline{s}b\underline{\alpha(x)}\underline{\$}|+|\underline{h}|)}}{(O(1)|S||\underline{s}b\underline{\alpha(x)}\underline{\$}||\underline{h}|)^2} \geq \frac{1}{p(|x|)}2^{-|\alpha(x)|},$$

where $p$ is a polynomial, dominates $\mu(x)$. □

## Acknowledgments

The authors are very grateful to Yuri Matiyasevich and Dmitry Itsykson for helpful comments.

## References


[AD00] Sergei I. Adian and Valery G. Durnev. Decision problems for groups and semigroups. *Russian Mathematical Surveys*, 55(2):207–296, 2000.

[BG95] Andreas Blass and Yuri Gurevich. Matrix transformation is complete for the average case. *SIAM Journal on Computing*, 24(1):3–29, 1995.

[BT06] Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1):1–106, 2006.

[Gur91] Yuri Gurevich. Average case completeness. *Journal of Computer and System Sciences*, 42(3):346–398, 1991.

[Lev86] Leonid A. Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986.

[Lev03] Leonid A. Levin. The tale of one-way functions. *Problems of Information Transmission*, 39(1):92–103, 2003.

[Mya07] Alexei Myasnikov. Generic complexity of undecidable problems. In *CSR'07*, pages 407–417, 2007.

[Tse56] Grigory S. Tseitin. Associative calculus with insoluble equivalence problem. *Dokl. Akad. Nauk SSSR*, 107:370–371, 1956. In Russian.

[VR92] Ramarathnam Venkatesan and Sivaramakrishnan Rajagopalan. Average case intractability of matrix and diophantine problems. In *STOC'92*, pages 632–642, 1992.

[Wan97] Jie Wang. Average-case intractible np problems. *Advances in Languages, Algorithms, and Complexity*, pages 313–378, 1997.

[Wan99] Jie Wang. Distributional word problem for groups. *SIAM Journal on Computing*, 28(4):1264–1283, 1999.

[WB95] Jie Wang and Jay Belanger. On the np-isomorphism problem with respect to random instances. *Journal of Computer and System Sciences*, 50(1):151–164, 1995.