1 Introduction

Theorem 1. *Let X be a nonempty subset of $|\mathcal{A}|$. The substructure $[X]_{\mathcal{A}}$ generated by X in $\mathcal{A}$ is the set V of all the values of all the L-terms for all the states over X for L-structure $\mathcal{A}$.*

Proof. Let $a \in [X]_{\mathcal{A}}$. It means that a belongs to any substructure containing X. To prove that $a \in V$ it is enough to prove that V is a substructure.

If c is a constant, the value of c in $\mathcal{A}$ is in V by definition as the value is $\sigma(c)$ for any state σ.

Let f be a signature operation symbol of arity n. Let $b_1,\dots, b_n$ be token from V. By definition, for $i = 1, \dots, n$, b_i is $\sigma_i(t_i)$ for an L-term t_i and a state σ_i over X for L-structure $\mathcal{A}$. Without loss of generality, we can suppose that for different i and j from $\{1, \dots, n\}$, t_i and t_j contain no common variable. Indeed, if a variable x occurs both in t_i and in t_j, we choose a new variable z such that $t_1,\dots, t_n$ do not contain z, produce the change from x to z in t_i, and put $\sigma_i(z) = \sigma_i(x)$. Now for variable x occurring in t_i where $i \in \{1, \dots, n\}$, we put $\sigma(x) = \sigma_i(x)$. Otherwise we put $\sigma(x) = \sigma_1(x)$. Then σ is a state over X and

$$\sigma(f(t_1, \dots, t_n)) = f^{\mathcal{A}}(\sigma(t_1), \dots, \sigma(t_n)) =$$
$$f^{\mathcal{A}}(\sigma_1(t_1), \dots, \sigma_n(t_n)) = f^{\mathcal{A}}(b_1, \dots, b_n).$$

So $f^{\mathcal{A}}(b_1, \dots, b_n)$ lies in V.

We used that $\sigma(t_i) = \sigma_i(t_i)$. The equality can be easy proved by induction on the number of operation symbols in t_i. It holds for variables occurring in t_i by definition of σ_i. Moreover, if t_i is $g(s_1, \dots, s_k)$,

$$\sigma(g(s_1, \dots, s_k)) = g^{\mathcal{A}}(\sigma(s_1), \dots, \sigma(s_k)) =$$
$$g^{\mathcal{A}}(\sigma_i(s_1), \dots, \sigma_i(s_k)) = \sigma_i(g(s_1, \dots, s_k)).$$

Thus V is closed under any signature operation and is a substructure.

In another direction, let $a \in V$. It means that a is the value of an L-term for a state over X for $\mathcal{A}$. Using induction on the number i of the operation symbols in the term, we prove that $a \in [X]_{\mathcal{A}}$.

If $i = 0$, the term is a variable and its value belongs to X by the definition of a state over X. If the term is constant c, a is $c^{\mathcal{A}}$ and a belongs to any substructure of $\mathcal{A}$ by definition. If a is the value of term t and t is $f(t_1, \dots, t_n)$, a is $f^{\mathcal{A}}(\sigma(t_1), \dots, \sigma(t_n))$ where σ is a state over X. By induction, $b_i = \sigma(t_i) \in [X]_{\mathcal{A}}$ for $i = 1,\dots, n$. As $[X]_{\mathcal{A}}$ is closed under any signature operation, $a = f^{\mathcal{A}}(b_1, \dots, b_n) \in [X]_{\mathcal{A}}$. $\square$

Let E be a congruence for L-structure $\mathcal{A}$. The factor-structure $\mathcal{A}$ modulo E is L-structure and is defined by the following way. The support of $\mathcal{A}$ modulo E is the set of all the equivalence classes for E. For each signature relation symbol P of arity n and any $a_1,\dots, a_n$ from $|\mathcal{A}|$, $(a_1/E, \dots, a_n/E) \in P^{\mathcal{A}/E}$ iff $(a_1, \dots, a_n) \in P^{\mathcal{A}}$ where $\mathcal{A}/E$ is $\mathcal{A}$ modulo E and for $a \in |\mathcal{A}|$, a/E is the equivalence class of E containing a.

The correctness of the definition is followed by the congruence definition.

Lemma 2. *Let Φ_1 is obtained from L-sentence Φ by replacement $t_1 = t_2$ with $E(t_1, t_2)$ for any L-terms t_1 and t_2. The equality symbol $=$ does not occur in Φ_1 and Φ_1 is an $\langle L, E^{(2)}\rangle$-formula. We suppose that E does not belong to L. Let $E^{\mathcal{A}}$ be a congruence for L-structure $\mathcal{A}$.*

$\mathcal{A}/E^{\mathcal{A}} \models \Phi$ *iff* $(\mathcal{A}, E^{\mathcal{A}}) \models \Phi_1$.

Proof. By induction on the number of propositional connectives and quantifiers in L-formula Ψ, we prove for any state s for $(\mathcal{A}, E^{\mathcal{A}})$,

$s/E^{\mathcal{A}} \models \Psi$ iff $s \models \Psi_1$

where $s/E^{\mathcal{A}}(x) = (s(x))/E^{\mathcal{A}}$ for any variable x. Ψ_1 is obtained from L-formula Ψ by replacement $t_1 = t_2$ with $E(t_1, t_2)$ for any L-terms t_1 and t_2. $\square$

2 Definitions

To have a possibility to range languages by complexity, we need a notion of Turing machines.

A Turing machine has the sole tape. There is a finite input alphabet. Its symbols are written in the tape cells. In any moment, any cell contains the only symbol. Before beginning of working, the tape presents an input word.

For any Turing machine, there are a finite set of states. In any moment, the Turing machine is in one of the states and examines one of the tape cells. The examined symbol is the symbol written in the examined cell.

In the first moment, any Turing machine examines the initial cell of its tape having the initial state. Any cell can be chosen as the initial cell.

To examine a cell on its tape and to be in a state, depending on its rule for the examined symbol and the state, a Turing machine changes the symbol in the examined cell of the tape, produces a movement (to the left, to the right, or no movement) on its tape, and move to a new state.

If the new state is final, the Turing machine stops. Otherwise, it continues to work according to its rules.

It is enough of general descriptions. Let us move to the formal definitions.

Definition 1. *A Turing machine (in short, TM) M is a collection of following objects:*

1. *a finite tape (or input, or external) alphabet Σ containing so-called "empty" symbol $\wedge$ and, at least, one more symbol, for example, $|$,*
2. *a finite state (or internal) alphabet Q such that $(Q \cap \Sigma) = \emptyset$,*
3. *an initial state $q_1 \in Q$ and final state $! \notin (Q \cup \Sigma)$,*
4. *a mapping M from $(\Sigma \times Q)$ in $(\Sigma \times \{L, R, C\} \times (Q \cup \{!\}))$.*

Symbols from $\{L, R, C\})$ are the movement directions. L is "left", R is "right", and C is "stop".

Any Turing machine M can be seen as a table with $|\Sigma|$ columns marked by elements of Σ and $|Q|$ rows marked by elements of Q where $|A|$ denotes the number of the elements of a finite set A. In the table, the cell, situated simultaneously in the column marked by s and in the row marked by q, is $M(s, q)$. Anyway, we suppose that the initial state marks the first row.

For example, the following table

Right$_*$	$\wedge$	$\vert$	$*$	1	2	3
q	Rq_1	Rq_1	Rq_1	Rq_1	Rq_1	Rq_1
q_1	R	R	!	R	R	R

represents a Turing machine with six input symbols $\wedge$, $|$, $*$, 1, 2, and 3 and two states q which is initial and q_1.

In a presentation of a Turing machine as a table, in the cells, we usually omit the new state if the state does not change, the new tape symbol if the symbol does not change and the movement direction if the direction is C (stop).

So, in the example, the table really is

Right$_*$	$\wedge$	$\vert$	$*$	1	2	3
q	$\wedge Rq_1$	$\vert Rq_1$	$*Rq_1$	$1Rq_1$	$2Rq_1$	$3Rq_1$
q_1	$\wedge Rq_1$	$\vert Rq_1$	*C!	$1Rq_1$	$2Rq_1$	$3Rq_1$

We use term "word" in the common sense. Remember the meaning. Any symbol from D is a word in alphabet D. Symbol λ of the empty word is a word in alphabet D. If a and b are words in D then ab also is a word in D. D^* denotes the set of all the words in alphabet D.

To describe the work of a Turing machine, we need a symbol to point out the beginning and the end of a tape. We use h for the symbol. We suppose $h \notin (\Sigma \cup Q \cup \{!\})$.

Turing machines transform some words in $(\Sigma \cup Q \cup \{h, !\})$ into words in the alphabet.

Definition 2. *A Post word (in short, PW) for TM M with state alphabet Q and input alphabet Σ is a word $hArsBh$ where A and B are words in Σ, r is from $Q \bigcup \{!\}$, and s is from Σ. It may happen that A or B is empty, or both of them are empty. But the first symbol of A and the last symbol of B are not $\wedge$. If r is the initial state, the Post word is called initial. If r is the final state, the Post word is called final.*

Turing machines transform Post words into Post words.

Definition 3. *Fix input alphabet Σ. Let M be a TM with state alphabet Q and Π be a Post word for M.*

If Π is a final PW, then $M(\Pi)$ is Π.

Let Π be $hArsBh$ where r is from Q.

Case 1. $M(s, r) = s'Lr'$, s' is not empty symbol, A is A_1a, a is from Σ, A_1 is from Σ^*.

Then $M(\Pi)$ is $hA_1r'as'Bh$.

Case 2. $M(s, r) = s'Lr'$, s' is not empty symbol, A is the empty word.

Then $M(\Pi)$ is $hr' \wedge s'Bh$.

Case 3. $M(s,r) = \wedge Lr'$, *A is* A_1a, *a is from* Σ, A_1 *is from* Σ^*, *B is not the empty word.*

Then $M(\Pi)$ *is* $hA_1r'a\wedge Bh$.

Case 4. $M(s,r) = \wedge Lr'$, *A is* A_1a, *a is from* Σ, A_1 *is from* Σ^*, *B is the empty word.*

Then $M(\Pi)$ *is* $hA_1r'ah$.

Case 5. $M(s,r) = \wedge Lr'$, *A is the empty word, B is not the empty word.*

Then $M(\Pi)$ *is* $hr'\wedge\wedge Bh$.

Case 6. $M(s,r) = \wedge Lr'$, *A and B are the empty words.*

Then $M(\Pi)$ *is* $hr'\wedge h$.

Case 7. $M(s,r) = s'Rr'$, s' *is not empty symbol, B is* bB_1, *b is from* Σ, B_1 *is from* Σ^*.

Then $M(\Pi)$ *is* $hAs'r'bB_1h$.

Case 8. $M(s,r) = s'Rr'$, s' *is not empty symbol, B is the empty word.*

Then $M(\Pi)$ *is* $hAs'r'\wedge h$.

Case 9. $M(s,r) = \wedge Rr'$, *B is* bB_1, *b is from* Σ, B_1 *is from* Σ^*, *A is not the empty word.*

Then $M(\Pi)$ *is* $hA\wedge r'bB_1h$.

Case 10. $M(s,r) = \wedge Rr'$, *B is* bB_1, *b is from* Σ, B_1 *is from* Σ^*, *A is the empty word.*

Then $M(\Pi)$ *is* $hr'bB_1h$.

Case 11. $M(s,r) = \wedge Rr'$, *B is the empty word, A is not the empty word.*

Then $M(\Pi)$ *is* $hA\wedge r'\wedge h$.

Case 12. $M(s,r) = \wedge Rr'$, *A and B are the empty words.*

Then $M(\Pi)$ *is* $hr'\wedge h$.

Case 13. $M(s,r) = s'Cr'$.

Then $M(\Pi)$ *is* $hAr's'Bh$.

We say M transforms Post word Π *into* $M(\Pi)$, *by one step, and we write:* $\Pi \Longrightarrow^1_M M(\Pi)$. *If* $\Pi \Longrightarrow^1_M \Pi_1$ *and* $\Pi_1 \Longrightarrow^n_M \Pi_2$, *then* $\Pi \Longrightarrow^{n+1}_M \Pi_2$. $\Pi \Longrightarrow^+_M \Pi_1$ *denotes that there exists some positive natural i such that* $\Pi \Longrightarrow^i_M \Pi_1$. *In this case, we say that M transforms* Π *into* Π_1. $\Pi \Longrightarrow^*_M \Pi_1$ *denotes that either* $\Pi \Longrightarrow^+_M \Pi_1$ *or* Π_1 *is* Π.

Roughly speaking, although the tape is infinite, the set of all the nonempty cells is finite in the first moment, and the set remains finite in any moment. The situation is described by Post words.

In any moment, the only cell is examined. R is the one cell movement to the right, L is the one cell movement to the left, C is no movement. A Post word is the concatenation of the tape word situated before the examined cell, the state, the examined symbol (mean, the content of the examined cell), and the tape word situated after the examined cell. Any cell contains the only symbol. The symbol is external.

First at all, the examined symbol is replaced with the new one defined by the machine rule for the pair (the examined symbol, the active state), then, depending on the movement direction in the rule, either there produces one cell movement to the left or to the right, or there produces no movement, and, finally, the active state is replaced with the new one defined by the rule. In the first moment, the active state is the initial state.

To obtain the next Post word, in some cases, we have to remove an end empty symbol and we have to add the examined empty symbol. To be correct, if we examine the last tape symbol and we are told by the rule to move to the right, we have to add the empty symbol as the new last tape symbol to examine it at the next step. Symmetrically, if we examine the first tape symbol and we are told by the rule to move to the left, we have to add the empty symbol as the new first tape symbol to examine it at the next step.

Example 1. Let us consider TM Right_* again.

Right_* has two rows marked by states q and q_1. In the second row, one cell is !. It is the cell for the column marked by $*$. Other cells are R. In the first row, all the cells are Rq_1. State q is the initial state.

Let us consider PW $\Pi_0 \;=\; hq\wedge|\,*1||\,*h$ for Right_*.

As $\text{Right}_*(\wedge, q)$ is $\wedge Rq_1$, case 10 holds where B is $|\,*1||\,*$.

TM Right_* transforms Π_0 into $\Pi_1 = \text{Right}_*(\Pi_0)$, which is, by definition for case 10, $hq_1|\,*1||\,*h$.

As $\text{Right}_*(|\,, q_1)$ is $|\,Rq_1$, case 7 takes place where A is the empty word, and B is $*1||*$.

By definition for case 7, Π_1 is transformed into Π_2 which is $h|\,q_1*1||\,*h$

As $\text{Right}_*(*, q_1)$ is $*C!$, Post word Π_2 is transformed into $h|\,!*1||\,*h$.

Thus,

$$hq\wedge|\,*1||\,*h \Longrightarrow^3_{\text{Right}_*} h|\,!*1||\,*h.$$

Post word $h|\,!*1||\,*h$ is final.

Example 2. Let us consider TM

Bad	$\wedge$	$\vert$	$*$	1	2	3
q	Rq_1	Rq_1	Rq_1	Rq_1	Rq_1	Rq_1
q_1	R	L	!	R	R	R

Let us consider PW $\Pi_0 \;=\; hq\wedge|\,*1||\,*h$ for Right_*.

As $\text{Bad}(\wedge, q)$ is $\wedge Rq_1$, case 10 holds where B is $|\,*1||\,*$.

TM Bad transforms Π_0 into $\Pi_1 = \text{Right}_*(\Pi_0)$, which is, by definition for case 10, $hq_1|\,*1||\,*h$.

As $\text{Bad}(|\,, q_1)$ is $|\,Lq_1$, case 2 takes place where A is the empty word, and B is $*1||*$.

By definition for case 2, Π_1 is transformed into Π_2 which is $hq_1\wedge|\,*1||\,*h$

As $\text{Bad}(\wedge, q_1)$ is $\wedge Rq_1$, Post word Π_2 is transformed into $hq_1|\,*1||\,*h$ which is Π_1 again.

Thus,

$$\Pi_0 \Longrightarrow^n_{\text{Bad}} \Pi_i$$

where i is the remainder modulo 2 of $n > 0$. It is obvious, beginning from Π_0, MT Bad never reaches any final Post word.

Definition 4. *Let Π_0 is an initial Post word for TM M. M stops on Π_0 iff there are a final Post word Π_1 for M and a positive natural number n such that*

$$\Pi_0 \Longrightarrow_M^n \Pi_1.$$

Exercise 1. *For any positive natural number n, a mapping f from a subset X of*

$$\omega^n = \{(a_1, \dots,\ a_n) \mid a_1, \dots,\ a_n \in \omega\}$$

to ω is called an arithmetical function of arity n. X is called the domain of f and is denoted $DOM(f)$. TM M calculates arithmetical function f of arity n iff for any initial Post word $hq \wedge |^{a_1} * \cdots * |^{a_n} * h$ where $a_1, \dots,\ a_n \in \omega$, M stops on the Post word iff $(a_1, \dots,\ a_n) \in DOM(f)$, and if M stops on the Post word, there is a positive natural number n such that

$$hq \wedge |^{a_1} * \cdots * |^{a_n} * h \Longrightarrow_M^n h! \wedge |^{f(a_1, \dots,\ a_n)} * h.$$

If $f(a_1, \dots,\ a_n) = 0$, the final Post word is $h! \wedge *h$. An arithmetical function is called computable iff there is a Turing machine computing the arithmetical function.

Construct Turing machines computing arithmetical functions addition, multiplication and so on. For instance, construct a Turing machine computing for any natural numbers $n > 0$ and $m > 1$, the natural number i such that m^i divides n (there is a natural number k such that $n = k \times m^i$), and m^{i+1} does not divide n.

Suppose $| \in \Sigma$. A Turing machine with external alphabet Σ itself can be viewed as a word in alphabet $(\{q, R, L, C\} \cup \Sigma)$ if we identify the machine and its table, represent states as $|^i q$ for some natural numbers i, represent q as the initial state, represent any row as the sequence of its mark and its cells, and represent the Turing machine as the sequence of its rows.

For example, Right$*$ is represented by word

$$q \wedge R|q|R|q * R|q1R|q2R|q3R|q|q \wedge R|q|R|q * C!1R|q2R|q3R|q.$$

Thus, we need four additional symbols only to present any Turing machine with the external alphabet Σ.

In the presentation of a Turing machine, we replace $\wedge$ with $|\wedge\wedge$ and replace $|$ with $|||$. The next step is to replace additional symbols with, for example, $\wedge||$ (it is the substitution for q), $\wedge|\wedge$, $\wedge\wedge|$, and $\wedge\wedge\wedge$, After the substitutions, any Turing machine with the external alphabet Σ is a word in alphabet Σ.

Point out that any Turing machine representation was finished by q. After the substitutions, any Turing machine representation is finished by $|$.

For any word $A \in \Sigma^*$ which last symbol is $|$, initial Post word $P(A)$ is $hq \wedge Ah$.

Theorem 3. *There is no TM M which satisfies the following conditions:*

1. *if A is a representation of a Turing machine, M stops on $P(A)$ examining a cell with $\wedge$ if the Turing machine presented by A stops on $P(A)$;*

2. *if A is a representation of a Turing machine, M stops on $P(A)$ examining a cell with $|$ if the Turing machine presented by A does not stop on $P(A)$.*

Proof. Suppose M is such a Turing machine. Improve M to obtain a new Turing machine N by the following way. N has two new states r and r'. We substitute a new state r for the final state in all the rules of M, include the obtained rules in the rules of N and add rules $N(\wedge, r) = (\wedge, R, r')$, $N(|, r) = (|, C, !)$, and $N(s, r') = (s, R, r')$ for all $s \in \Sigma$. So, if q is a state of M and $s \in \Sigma$, $N(s, q) = M(s, q)$ if $M(s, q)$ is not the final state. Otherwise, $N(s, q) = r$. No matter what N is doing in other cases.

Thus, if A is a representation of a Turing machine which stops on $P(A)$, M stops on $P(A)$ examining a cell with $\wedge$, and N moves to r' and never stops. If A is a representation of a Turing machine which does not stop on $P(A)$, M stops on $P(A)$ examining a cell with $|$, and N stops.

Let A be the representation of N.

If N stops on $P(A)$, A is a representation of a Turing machine which stops on $P(A)$, and N does not stop on $P(A)$. A contradiction.

If N does not stop on $P(A)$, A is a representation of a Turing machine which does not stop on $P(A)$, and N stops. A contradiction again. □

Post word $hq \wedge h$ where q is the initial state is called the empty initial Post word.

Theorem 4. *There is no TM M which satisfies the following conditions:*

1. *if A is a representation of a Turing machine, M stops on $P(A)$ examining a cell with $\wedge$ if the Turing machine presented by A stops on the empty initial Post word;*

2. *if A is a representation of a Turing machine, M stops on $P(A)$ examining a cell with $|$ if the Turing machine presented by A does not stop on the empty initial Post word.*

Proof (not absolutely formal). Having a representation A of a Turing machine, we construct the representation B of a new Turing machine which work on the empty Post word is as following. The new TM constructs the given representation from the empty tape, goes to the empty cell situated immediately before the constructed representation and then works as the TM presented by A does.

Thus, the Turing machine presented by A stops on initial Post word $P(A)$ iff the Turing machine presented by B stops on the empty initial Post word.

We left to our reader to describe in details a Turing machine M_0 which produces for any given representation A of a Turing machine, a transformation of initial Post word $P(A)$ to the final Post word $h! \wedge Bh$.

If there is a TM M satisfying the conditions of the theorem, the composition (the consecutive work of) M_0 and M satisfies the conditions of Theorem 3.

To obtain the composition, one can suppose that M_0 and M have no common states, substitute the initial state of M for the final state of M_0 in all the rules of M_0, and join all the rules of the Turing machine obtained from M_0 by the reorganization and all the rules of M. □

A modification of the Turing machines is so-called the *one-sided* Turing machines (in short, 1-TM).

No difference in the definition of Turing machines itself. But there is an essential difference in the definition of a Post word. A Post word is $hArsBh$ where r is a state or the final state, s is a symbol from the external alphabet Σ, A and B are words from Σ^*. It may happen that A or B is empty, or both of them are empty. But the last symbol of B are not $\wedge$.

The tape has the beginning (or the first cell) now. A present the part of the tape from the beginning to the examined cell.

The difference implies differences in the definition of the Post word transformation.

The first difference is that a Post word contains the whole of the tape before the examined cell. It is possible because this part of the tape is finite. So cases 2, 5, 6, 10, 12 do not take place. The consideration of other cases is the same as for the two-sided TM. We omit details.

But there is an additional case. An one-sided Turing machine M can examine in a state q the first cell containing a symbol s and can have rule $M(s,q) = (s', L, q')$ for some s' and q'. An one-sided Turing machine M is called coming down in time of its work if the machine examines the first cell containing s in a state q such that $M(s,q) = (s', L, q')$ for a state q' and a symbol s'. In such a case, the next Post word does not exist.

For example, Right$_*$ and Bad can be viewed as 1-TM.

The examples 1 and 2 is also good for new 1-TM Right$_*$ and Bad.

The definitions of a representation of a Turing machine and the Post word $P(A)$ for a representation A need no corrections to be apply to 1-TM. Of course, the empty initial Post word continue to be $hq \wedge h$ where q is the initial state.

After the corrections, Theorems 3 and 4 still hold and have the same proofs as for the two-sided Turing machines.

We will restrict ourself to consideration one-sided Turing machines only.

Definition 5. *Consider a set $\mathfrak{U}$ of words in $(\Sigma \setminus \{\wedge\})$. The set is called recursive iff there is 1-TM M such that for any $A \in (\Sigma \setminus \{\wedge\})^*$, M stops on $hq \wedge Ah$ seeing $\wedge$ if $A \in \mathfrak{U}$, and M stops on $hq \wedge Ah$ seeing an external symbol different from $\wedge$ if $A \notin \mathfrak{U}$.*

Definition 6. *Consider a set $\mathfrak{U}$ of words in $(\Sigma \setminus \{\wedge\})$. The set is called recursively enumerable (in short, r.e.) iff there is 1-TM M such that for any $A \in (\Sigma \setminus \{\wedge\})^*$, M stops on $hq \wedge Ah$ seeing $\wedge$ if $A \in \mathfrak{U}$, and M does not stop on $hq \wedge Ah$ if $A \notin \mathfrak{U}$.*

It is a common usage to say that there is an algorithm to answer as to belonging to $\mathfrak{U}$ iff $\mathfrak{U}$ is recursive.

Theorem 5 (Post). *A set $\mathfrak{U}$ of words in $(\Sigma \setminus \{\wedge\})$ is recursive iff sets $\mathfrak{U}$ and $((\Sigma \setminus \{\wedge\})^* \setminus \mathfrak{U})$ both are recursively enumerable.*

Proof. Suppose M is such a Turing machine which witnesses the recursiveness of $\mathfrak{U}$. Improve M to obtain a new Turing machine N by the following way. N

has two new states r and r'. We substitute a new state r for the final state in all the rules of M, include the obtained rules in the rules of N and add rules $N(\wedge, r) = (\wedge, C, !)$, $N(s, r) = (s, R, r')$ for all s from Σ if s is different from $\wedge$, and $N(s, r') = (s, R, r')$ for all $s \in \Sigma$. So, if q is a state of M and $s \in \Sigma$, $N(s, q) = M(s, q)$ if $M(s, q)$ is not the final state. Otherwise, $N(s, q) = r$. No matter what N is doing in other cases.

N witnesses that $\mathfrak{U}$ is recursively enumerable. The recursive denumerability of the complement of $\mathfrak{U}$ has a similar proof.

Suppose sets $\mathfrak{U}$ and $((\Sigma \setminus \{\wedge\})^* \setminus \mathfrak{U})$ both are recursively enumerable. Let TM M_1 witnesses that $\mathfrak{U}$ is recursively enumerable and TM M_2 witnesses the recursive enumeration of the complement of $\mathfrak{U}$.

First, we construct a Turing machine with two tapes. It works step-by-step. At odd steps, it works on the first tape as M_1 works. At even steps, it works on the second tape as M_2 works. It stops if it stops on a tape. It stops anyway because any word either is in a set or is in the complement of the set. If it stops on the first tape, the input is in $\mathfrak{U}$. Otherwise, the input is not in $\mathfrak{U}$. We left details to the reader.

There is a general procedure to reconstruct a two tape Turing machine in an one tape TM. A good idea is to introduce duplicates for any symbols. As tape is one-sided, any cell has the number. The first cell is numbered by 0, second one is numbered by 1, third one is numbered by 2, next one is numbered by 3 and so on. Then consider the pair of symbols composed from the symbols of the number i cells as the symbol in the number i cell of the new tape. To mark the examined cell, we replace its symbol with the duplicate of the symbol. The new TM moves to the right until it finds the examined cells both. Then it is returning to the beginning of the tape. In time of the return, it changes the symbols in the examined cells and marks the new examined cells. After the work, it move to the new state. We left details to the reader. □

Example 3. The natural numbers can be viewed as words in alphabet $\{|\}$. Let $\Sigma = \{\wedge, |\}$. The set of the even numbers are recursive. Indeed, consider the following 1-TM

Even	$\|$	$\wedge$
q	Rq_1	Rq_1
q_1	Rq_2	!
q_2	Rq_1	$\|!$

Even transforms $hq \wedge |^i h$ into $h \wedge |^i ! \wedge h$ if i is even.

Even transforms $hq \wedge |^i h$ into $h \wedge |^i ! \mid h$ if i is odd. So Even stops anyway. It stops seeing $\wedge$ iff i is even.

Exercise 2. *The image $IM(f)$ of an arithmetical function f is $\{f(\bar{x}) \mid \bar{x} \in DOM(f)$. Prove that the domain and the image of any computable function are recursively enumerable. See Exercise 1 for additional definitions.*

Exercise 3. *Give an example of a computable function which both the image and the domain are not recursive.*

Definition 7. *Two sets $\mathfrak{U}$ and $\mathfrak{V}$ of words from Σ^* are called separable iff there is a recursive set $\mathfrak{W}$ of words from Σ^* such that $\mathfrak{U} \subseteq \mathfrak{W}$ and $\mathfrak{V} \subseteq (\Sigma^* \setminus \mathfrak{W})$. We say that $\mathfrak{W}$ separates $\mathfrak{U}$ and $\mathfrak{V}$. $\mathfrak{U}$ and $\mathfrak{V}$ is called inseparable if they are not separable.*

Note 6. Any separable $\mathfrak{U}$ and $\mathfrak{V}$, of course, are disjoint. It is absolutely obvious that if $\mathfrak{U}$ and $\mathfrak{V}$ are simultaneously inseparable and disjoint, then each of them are not recursive. Indeed, if one of the sets is recursive, then the set or its compliment separates $\mathfrak{U}$ and $\mathfrak{V}$.

In time of its work on the empty Post word, an one-sided Turing machine examines cells. If the set of all the examined cells is finite, the 1-TM is called *cyclic*. If 1-TM M stops on the empty Post word, of course, M is cyclic.

If A is a representation of an one-sided Turing machine M, $T(A)$ denotes M itself.

Ideas token from the proofs of Theorem 4 and Theorem 3 are enough to prove the following

Theorem 7. *The set* Cyclic *of all the representations A of cyclic one-sided Turing machines and the set* Down *of all the representations of coming down on the empty Post word one-sided Turing machines are inseparable.*

Proof. By contradiction.

Let 1-MT M_2 stops anyway seeing either $\wedge$ or, for example, $|$, stops on $P(A)$ seeing $\wedge$ if $A \in$ Cyclic, and stops on $P(A)$ seeing $|$ if $A \in$ Down.

We reorganize M_2 into M_1 by adding new states r and r'. We substitute a new state r for the final state in all the rules of M_2, include the obtained rules in the rules of M_1 and add rules $M_1(\wedge, r) = (\wedge, L, r')$, $M_1(|, r) = (|, C, !)$, and $M_1(s, r') = (s, L, r')$ for all $s \in \Sigma$. So, if q is a state of M_2 and $s \in \Sigma$, $M_1(s, q) = M_2(s, q)$ if $M_2(s, q)$ is not the final state. Otherwise, $M_1(s, q) = r$. No matter what M_1 is doing in other cases.

It is obvious that M_1 stops on $P(A)$ if $A \in$ Down, and M_1 comes down the tape on $P(A)$ if $A \in$ Cyclic. Moreover, for any initial Post word, either M_1 stops on the initial Post word, or M_1 comes down on the initial Post word.

Now take a part of the Theorem 4 proof.

Having a representation A of a Turing machine, we construct the representation B of a new Turing machine which work on the empty Post word is as following. The new TM constructs the given representation from the empty tape, goes to the empty cell situated immediately before the constructed representation and then works as the TM presented by A does.

Thus, the Turing machine presented by A stops on initial Post word $P(A)$ iff the Turing machine presented by B stops on the empty initial Post word. Moreover, the Turing machine presented by A comes down the tape on initial Post word $P(A)$ iff $B \in$ Down.

We proposed to the reader to describe in details a Turing machine M_0 which produces for any given representation A of a Turing machine, a transformation of initial Post word $P(A)$ to the final Post word $h! \wedge Bh$.

Let us ask whether the composition M_3 of M_0 and M_1 comes down the tape on the empty initial Post word. By the definition of M_1, for any initial Post word, M_3 either comes down the tape on the empty initial Post word, or stops on the empty initial Post word. Let A be the representation of the composition.

If M_3 comes down the tape on the empty initial Post word, then M_1 stops on $P(A)$, and M_3 stops on the empty initial Post word.

If M_3 stops on the empty initial Post word, then M_1 comes down the tape on $P(A)$, and M_3 comes down the tape on the empty initial Post word.

A contradiction. $\square$

A generalization of a Turing machine is a nondeterministic Turing machine.

Definition 8. *A nondeterministic Turing machine (in short, N-TM) M is a collection of following objects:*

1. *a finite tape (or input, or external) alphabet Σ containing so-called "empty" symbol $\wedge$ and, at least, one more symbol, for example, $|$,*
2. *a finite state (or internal) alphabet Q such that $(Q \cap \Sigma) = \emptyset$,*
3. *an initial state $q_1 \in Q$ and final state $! \notin (Q \cup \Sigma)$,*
4. *a mapping M from $(\Sigma \times Q)$ in the set of the finite subsets of $(\Sigma \times \{L, R, C\} \times (Q \cup \{!\}))$.*

Roughly speaking, if a TM has exactly one rule for any pair of a state and an examined symbol, a N-TM for such a pair can have a number of rules. It is possible to choose any of appropriate rules. For example, $M(a, q)$ can be $\{(b, L, r), (d, R, s)\}$. Then, if M examines a cell with a in state q, M can either put b in the examined cell, move to the left and pass to state r, or put d in the examined cell, move to the right and pass to state s. The formal definition of Post words is the same, and the formal definition of the work of a N-TM on a Post word requires a correction. First, a choice of a rule has to be made. After that, the next Post word is the next Post word for a TM with the chosen rule.

3 Trakhtenbrot theorem

An L-sentence is called finitely valid iff it holds in any finite L-structure. An L-sentence is called finitely satisfiable iff it holds in a finite L-structure. An L-sentence is called finitely refutable iff its negation holds in a finite L-structure. An L-sentence is called inconsistent iff it has no models.

Theorem 8 (Trakhtenbrot). *The set of all the finitely valid sentences is not r.e.*

Note 9. It is nearly obvious that the compliment (to the all the words in the alphabet of the L-formulas) of the set of of all the finitely valid sentences is r.e. Indeed, having an enumeration of the finite L-structures, for an L-sentence, we double check the truth of the sentence in the considering structure. If the sentence holds, we come to consider the next finite L-structure. If the sentence is false, we stop the investigation and decide that the sentence belongs to the compliment. We left to the reader to construct an one-sided Turing machine made the described work. The construction is a kind of a very understandable but tiresome study.

By Post Theorem 5 and Note 9, Trakhtenbrot's Theorem is followed from the following

Lemma 10. *The set of finitely valid sentences is not recursive.*

Lemma 10 is followed from the following

Lemma 11. *The set of finitely satisfiable sentences is not recursive.*

Indeed, a sentence is finitely valid iff its negation is not finitely satisfiable.
We are going to prove a stronger

Theorem 12. *The sets of all the finitely satisfiable sentences and all the inconsistent sentences are inseparable.*

Indeed, by Note 6, any of inseparable pair sets is not recursive.

Proof. The proof is quite straightforward, although boring because of a lot of details. The presented proof is a modification of the well known Büchi's proof.

The idea is as follows. We use Theorem 7.

Now, given an one-sided Turing machine, we construct a formula describing the work of this Turing machine on the empty Post word. If the TM belongs to Cyclic, the formula is finitely satisfiable. If the TM belongs to Down, the formula is inconsistent.

If Theorem 12 does not hold, there is a set $\mathfrak{W}$ separating the sets of all the finitely satisfiable sentences and all the inconsistent sentences. We put an one-sided TM to $\mathfrak{V}$ iff the formula constructed for the 1-TM belongs to $\mathfrak{W}$. It is clear that $\mathfrak{V}$ is a recursive set separating Down and Cyclic. It contradicts to Theorem 7.

Namely, consider a (fixed for a fixed input alphabet Σ!) signature $\mathcal{T} = \langle P, \bar{S}, Q, L, R, C, 0, ' < \rangle$. Intuitively, we consider $\mathcal{T}$-structures such that the support is the set of all the natural numbers and $'$ is interpreted as the successor function.

Then, the binary relation P encodes the number of the examined cell of the Turing machine, as follows: $P(m, k)$ iff the kth cell of the tape on the mth step of the computation of the Turing machine on the (initially) empty tape is examined.

Similarly, the binary relation Q encodes the state of the Turing machine, as follows: $Q(m, k)$ represents that k is the internal state of the Turing machine on the mth step. Suppose that 0 is the final state and 1 is the initial state.

The sequence of the binary relations $\bar{S}(m, k)$ gives the symbol written in the kth cell when the mth step of the computation starts. For definiteness, $S_s(m, k)$

denotes that, when the mth step of the computation starts, the symbol s is written in the kth cell.

Finally, unary relations L, R, and C describe the movement direction on the step.

We can write an axiom for the "initial configuration", that is, that the tape is initially empty (all cells contain a special symbol "empty"), the head reads the first cell, and the state is initial.

Then an axiom describes general rules of the work of Turing machines.

Finally, using the program of the machine, we can write down an axiom for transition from the ith step to the $(i+1)$th step of the computation.

Also, one simple sentence claims that the Turing machine does not come down the tape.

This finite axiom system is finite satisfiable if the TM belongs to Cyclic, and is inconsistent if the TM belongs to Down.

To prove that, we observe some property of the axiom system.

If the Turing machine belongs to Down, it is obvious that the formula is false for models which support is the natural numbers and which ordering is the usual ordering of the natural numbers.

A crucial part of the proof is to prove that the constructed formulas has a model iff the formula has a model which support is the natural numbers and which ordering is the usual ordering of the natural numbers.

But if the TM belongs to Cyclic, it is obvious that the formula has a periodical model which support is the natural numbers and which ordering is the usual ordering of the natural numbers. A crucial part of the proof is to prove that the constructed formulas has such a periodical model iff the formula has a finite model.

It is time to move to details.

Axiom for the "initial configuration"(the initial axiom):

$$(P(0,0) \land \neg P(0,y\,') \land Q(0,0\,') \land S_{\land}(0,y)).$$

The axiom says that before its work, the Turing machine examines the first cell, has the initial state, and any cell contains the empty symbol.

Axiom for some properties of the work of any 1-TM (the property axiom):

$$(\bigwedge_{s\in\Sigma}(S_s(x,y) \to \bigwedge_{d\in(\Sigma\setminus\{s\})} \neg S_d(x,y))\land$$

$$\bigwedge_{s\in\Sigma}(\neg P(x,y) \to (S_s(x,y) \leftrightarrow S_s(x',y)))).$$

Here $\Phi \leftrightarrow \Psi$ is a shortening of

$$((\Phi \to \Psi) \land (\Psi \to \Phi)).$$

The axiom says that any cell in any moment contains the only symbol, and if a cell is not examined on the mth step, after the step, the cell contains the same symbol.

Additional axiom

$$\bigwedge_{i=0}^{q}(Q(x,i)\to \bigwedge_{j\in\{0,1,\ldots,q\},\ j\neq i}\neg Q(x,j))$$

where q is the number of the states of 1-TM M claims that in any moment, M is in the only state.

Move axiom

$$((L(x)\to (P(x,y')\leftrightarrow P(x',y)))\wedge$$
$$(R(x)\to (P(x',y')\leftrightarrow P(x,y)))\wedge$$
$$(C(x)\to (P(x',y)\leftrightarrow P(x,y))))$$

claims that

- if the Turing machine moves to the left, the number of the next examined cell decreases by 1
- if the Turing machine moves to the right, the number of the next examined cell increases by 1
- if the Turing machine does not move, the number of the next examined cell is equal to the number of the examined cell.

The transition axiom depends on the rules of the given 1-TM M. Axiom for transition from the xth step to the $(x+1)$th step:

$$(\bigwedge_{M(s,i)=(d,L,j)}((S_s(x,y)\wedge P(x,y)\wedge Q(x,i))\to (Q(x',j)\wedge L(x)\wedge S_d(x',y)))\wedge$$

$$\bigwedge_{M(s,i)=(d,R,j)}((S_s(x,y)\wedge P(x,y)\wedge Q(x,i))\to (Q(x',j)\wedge R(x)\wedge S_d(x',y)))\wedge$$

$$\bigwedge_{M(s,i)=(d,C,j)}((S_s(x,y)\wedge P(x,y)\wedge Q(x,i))\to (Q(x',j)\wedge C(x)\wedge S_d(x',y)))$$

The axiom says that the next state, the next contents of the examined cell and the next examined cell are defined by the corresponding rule of the Turing machine.

We need in a small correction of the transition axiom and of the move axiom. We add a new binary relation H and replace $P(x,y')$ with $H(x,y)$ and $P(x',y')$ with $H(x',y)$ in the transition axiom and in the move axiom. The axiom obtained by that way from the transition axiom is named the corrected transition axiom. The axiom obtained by that way from the move axiom is named the corrected move axiom.

In the initial axiom, we replace $P(0,y')$ with $H(0,y)$. Note that after the correction (and before the correction), the initial axiom contains no occurrence of x. We substitute x for 0 in the corrected initial axiom and obtain formula $\phi(x,x',y)$. Formula $(Z(x)\to\phi(x,x',y))$ is named the new initial axiom.

Let $\Psi(x, x', y)$ be the conjunction of the new initial axiom, the property axiom, the additional axiom, the corrected move axiom, the corrected transition axiom and formulas $(H(y, x) \leftrightarrow P(y, x'))$ and $\neg Z(x')$. Let $\Phi(0)$ is $(Z(0) \wedge (\forall x)(\forall y)\Psi(x, x', y))$. For instance, $\Phi(0)$ says that $Z(x')$ is false for any x. As $Z(0)$ holds, it means that 0 is not x' for any x.

Moreover, if the support of a model of $\Phi(0)$ is the natural numbers and $'$ denotes the next number, then $Z(x)$ holds iff $x = 0$, and the initial, property, additional, move and transition axioms hold.

It easy to prove by induction on the step number that if $\Phi(0)$ holds, the support is the natural numbers, $'$ denotes the next number, and the one-sided Turing machine works on the empty Post word, then the following properties hold:

1. $Q(m, k)$ denotes that k is the internal state of the Turing machine on the mth step,

2. $P(m, k)$ denotes that the kth cell of the tape on the mth step of the computation of the Turing machine is examined,

3. the sequence of the binary relations $\bar{S}(m, k)$ gives the symbol written in the kth cell when the mth step of the computation starts.

It proves that formula

$$(\Phi(0) \wedge (\forall y)(\forall x)(Z(y) \to \neg(L(x) \wedge P(x, y))))$$

is false in $(\omega,\ ', 0, P, H, \bar{S}, Q, Z, L, R, C)$ where $'$ and 0 have the usual interpretation for any interpretations $P, H, \bar{S}, Q, Z, L, R, C$ if the given 1-TM is in Down.

Moreover, the formula is finite satisfiable if the given 1-TM is in Cyclic.

Indeed, if the given 1-TM is in Cyclic, the formula has a model $(\omega,\ ', 0, P, H, \bar{S}, Q, Z, L, R, C)$ where $'$ and 0 have the usual interpretation, and there are natural numbers m and n such that for any binary signature relation U and any natural numbers $a > m$ and $b > m$, $(a, b) \in U$ iff $(a + n, b) \in U$ and $(a, b) \in U$ iff $(a, b + n) \in U$, and for any unary signature relation U and any natural number $a > m$, $a \in U$ iff $a + n \in U$.

So the binary relation E defined by rule

$$((x \leqslant m \vee y \leqslant m) \wedge x = y) \vee (x > m \wedge y > m \wedge x - y = 0 \text{ modulo } n))$$

is a congruence. The number of the congruence classes is finite. The formula contains no equality. By Lemma 2, the formula also holds in the factor-structure modulo E which is finite.

The last part of the proof is devoted to proving

Lemma 13. *If formula*

$$(\Phi(0) \wedge (\forall y)(\forall x)(Z(y) \to \neg(L(x) \wedge P(x, y))))$$

has a model then the formula has a model $(\omega,\ ', 0, P, H, \bar{S}, Q, Z, L, R, C)$ *where* $'$ *and* 0 *have the usual interpretation.*

Proof. Take a model of the formula. Consider the substructure of the model generated by $\{0\}$. By Theorem 1, the substructure is $\{0, 0', 0'', \dots\}$. As the formula is universal, the formula holds in the substructure. If the substructure is infinite, its reduct to signature $\langle ', 0\rangle$ is isomorphic to $(\omega, \ ', 0)$, and we are done.

Otherwise, there is an epimorphism τ from $(\omega, \ ', 0)$ to the substructure where $\tau(0) = 0$. For each binary signature relation U and any natural a and b, we put $U(a, b)$ to be true in $(\omega, \ ', 0)$ iff $U(\tau(a), \tau(b))$ holds in the substructure. Similarly, for each unary signature relation U and any natural a, we put $U(a)$ to be true in $(\omega, \ ', 0)$ iff $U(\tau(a))$ holds in the substructure. As the formula does not contain the equality symbol, it is routine to prove by induction on the number of propositional connectives in the formula that the formula holds in the obtained structure. $\square$

$\square$

Corollary 14 (Church). *The set of all the valid sentences is not recursive.*

Theorem 15 (Vaught). *Let $\Omega = \langle Q^{(3)}\rangle$. The sets of all the finitely satisfiable Ω-sentences and all the inconsistent Ω-sentences are inseparable.*

A sketch of a proof. Really, we proved Theorem 12 for a fix signature containing unary and binary relation names. We can introduce a ternary relation name R and interpret $R(a, u, v)$ or $R(a, u, u)$ as $Q_a(u, v)$ or $Q_a(u)$ depending on the arity of ath signature symbol Q_a. The argument reduces the general case to the case of one ternary relation name. $\square$

Remark 1.

1. It is well known and a proof can be found in any classical textbook of Mathematical Logic that the set of all the valid sentences is recursively enumerable.

2. Our proof does not use the equality symbol. Thus, really we proved stronger forms for Trakhtenbrot's Theorem and Theorem 12 for formulas without the equality symbol. The idea of the presented proof borrowed from J. Büchi.

3. Our proof uses a complex signature. So did the original proof of B.A. Trakhtenbrot (dated 1949), though the argument was in fact different. Later, however, the theorem was refined by R.L. Vaught (1960) to the case of any signature containing one binary symbol. But the reduction is not hard. There are many improvements of Theorem 12 in different directions. A.I.Malcev proposed to study such a kind of theorems for axiomatizable classes and carried Trakhtenbrot's Theorem over the class of the finite groups. Hao Wang, J. Büchi, Yu. Gurevich and other people studied of truth such a theorem for special sets of formulas.

Exercise 4. *Prove Note 9 formally.*

Exercise 5. *Prove for a signature L, that the sets of all the finitely refutable L-sentences (those that have a finite counter-model, a finite L-structure is a finite counter-model for an L-sentence iff the sentence is false in the structure) and all the valid L-sentences are inseparable. Prove the theorem for $L = \langle Q^{(3)} \rangle$. In other words, give a formal proof of Theorem 15.*

Exercise 6. *Prove for signature $L = \langle Q^{(2)} \rangle$, that the sets of all the finitely refutable L-sentences and all the valid L-sentences are inseparable.*

Hint: Consider two signatures $L_3 = \langle Q^{(3)} \rangle$ and $L_2 = \langle R^{(2)} \rangle$. Consider an L_3-structure $\mathcal{A}$. Let A be the support of $\mathcal{A}$. For any $a, b, c \in A$ such that $Q^{\mathcal{A}}(a, b, c)$, add new elements $d(a, b, c)$, $d_1(a, b, c)$, $d_2(a, b, c)$, $d_3(a, b, c)$, and put

$$R(d(a,b,c), d(a,b,c)),\ R(d_1(a,b,c), d_1(a,b,c)),$$

$$R(d_2(a,b,c), d_2(a,b,c)),\ R(d_3(a,b,c), d_3(a,b,c)),$$

$$R(a, d(a,b,c)),\ R(b, d_1(a,b,c)),\ R(c, d_2(a,b,c)),$$

$$R(d_1(a,b,c), d(a,b,c)),\ R(d_2(a,b,c), d_3(a,b,c)),\ R(d_3(a,b,c), d(a,b,c)).$$

Suppose the new elements are pairwise different and are different for different triple (a, b, c).

Exercise 7. *Remember that $LO(<)$ denotes the conjunction of the axioms of transitiveness, skew symmetry and totality. Prove that the sentence*

$$((LO(<) \land (\exists x)(\forall y) y \leqslant x) \lor \neg LO(<))$$

is finitely valid. Is the sentence valid?

Exercise 8. *Prove that formula*

$$(\exists x)(\forall y)(\exists z)((P(y,z) \to P(x,z)) \to (P(x,x) \to P(y,x)))$$

is finitely valid but is not valid.

Exercise 9. *Is the set of refutable L-sentences (those that have a counter-model) r.e. for any signature L?*

Exercise 10. *Prove for a signature L, that the class of all finite L-structures is not axiomatizable.*

Exercise 11. *Prove that the first-order theory of all finite graphs is not recursive.*

Exercise 12. *Prove that the first-order theory of all finite symmetric graphs is not recursive.*

Exercise 13. *Prove that the first-order theory of all finite reflexive graphs is not recursive.*

Exercise 14. *Prove that the first-order theory of all finite transitive graphs is not recursive.*

Exercise 15. *Prove that the theory of natural numbers with the successor function is recursive.*

Exercise 16. *The theory of natural numbers with $+$ and $<$ is called* Presburger Arithmetic. *Prove that Presburger Arithmetic is recursive.*