IAENG International Journal of Computer Science, 39:3, IJCS_39_3_01

The c-Fragment Longest Arc-Preserving Common Subsequence Problem

Anna Gorbenko, Vladimir Popov

***Abstract*—Arc-annotated sequences are useful in representing the structural information of RNA and protein sequences. In particular, arc-annotated sequences are useful in describing the secondary and tertiary structures of RNA and protein sequences. Structure comparison for RNA and for protein sequences has become a central computational problem bearing many challenging computer science questions. The longest arc-preserving common subsequence problem has been introduced as a framework for studying the similarity of arc-annotated sequences. It is a sound and meaningful mathematical formalization of comparing the secondary structures of molecular sequences. In this paper, we consider two special cases of the longest arc-preserving common subsequence problem, c-fragment LAPCS (unlimited, plain), c-fragment LAPCS (unlimited, unlimited). In particular, we consider a parameterized version of the 1-fragment LAPCS (unlimited, plain) problem, parameterized by the length l of the desired subsequence. We show $W[1]$-completeness of the problem. Also, we describe an approach to solve c-fragment LAPCS (unlimited, unlimited). This approach is based on constructing logical models for the problem.**

***Index Terms*—arc-annotation, longest common subsequence, parameterized complexity, $W[1]$-complete, logical models.**

I. Introduction

SEQUENCE-LEVEL investigation has become essential in modern molecular biology. But to consider genetic molecules only as long sequences consisting of the 4 basic constituents is too simple to determine the function and physical structure of the molecules. Additional information about the sequences should be added to the sequences. Early works with these additional information are primary structure based, the sequence comparison is basically done on the primary structure while trying to incorporate secondary structure data [1], [2]. This approach has the weakness that it does not treat a base pair as a whole entity. Recently, an improved model was proposed [3], [4].

Arc-annotated sequences are useful in describing the secondary and tertiary structures of RNA and protein sequences. See [3], [5]–[8] for further discussion and references. Structure comparison for RNA and for protein sequences has become a central computational problem bearing many challenging computer science questions. In this context, the longest arc preserving common subsequence problem (LAPCS) recently has received considerable attention [3]–[5], [8]–[13]. It is a sound and meaningful mathematical formalization of comparing the secondary structures of molecular sequences.

Ural Federal University, Department of Intelligent Systems and Robotics of Mathematics and Computer Science Institute, 620083 Ekaterinburg, Russian Federation. Email: gorbenko.aa@gmail.com, Vladimir.Popov@usu.ru

The work was partially supported by Analytical Departmental Program "Developing the scientific potential of high school".

II. Preliminaries and problem definitions

Given two sequences S and T over some fixed alphabet Σ, the sequence T is a subsequence of S if T can be obtained from S by deleting some letters from S. Notice that the order of the remaining letters of S bases must be preserved. The length of a sequence S is the number of letters in it and is denoted as $|S|$. For simplicity, we use $S[i]$ to denote the ith letter in sequence S, and $S[i, j]$ to denote the substring of S consisting of the ith letter through the jth letter.

Given two sequences S_1 and S_2 (over some fixed alphabet Σ), the classic longest common subsequence problem asks for a longest sequence T that is a subsequence of both S_1 and S_2.

An arc-annotated sequence of length n on a finite alphabet Σ is a couple $A = (S, P)$ where S is a sequence of length n on Σ and P is a set of pairs (i_1, i_2), with $0 < i_1 < i_2 < n+1$. In this paper we will then call an element of S a base. A pair $(i_1, i_2) \in P$ represents an arc linking bases $S[i_1]$ and $S[i_2]$ of S. The bases $S[i_1]$ and $S[i_2]$ are said to belong to the arc (i_1, i_2) and are the only bases that belong to this arc.

Given two annotated sequences S_1 and S_2 with arc sets P_1 and P_2 respectively, a common subsequence T of S_1 and S_2 induces a bijective mapping from a subset of $\{1, \ldots, |S_1|\}$ to subset of $\{1, \ldots, |S_2|\}$. The common subsequence T is arc-preserving if the arcs induced by the mapping are preserved, i.e., for any (i_1, j_1) and (i_2, j_2) in the mapping, $(i_1, i_2) \in P_1 \Leftrightarrow (j_1, j_2) \in P_2$.

The longest arc-preserving common subsequence problem (LAPCS) is to find a longest common subsequence of S_1 and S_2 that is arc-preserving (with respect to the given arc sets P_1 and P_2) [3].

LAPCS:

INSTANCE: An alphabet Σ, annotated sequences S_1 and S_2, $S_1, S_2 \in \Sigma^*$, with arc sets P_1 and P_2 respectively.

QUESTION: Find a longest common subsequence of S_1 and S_2 that is arc-preserving.

The arc structure can be restricted. We consider the following four natural restrictions on an arc set P which are first discussed in [3]:

1. no sharing of endpoints:
$\forall (i_1, i_2), (i_3, i_4) \in P$, $i_1 \neq i_4, i_2 \neq i_3$, and $i_1 = i_3 \Leftrightarrow i_2 = i_4$.
2. no crossing:
$\forall (i_1, i_2), (i_3, i_4) \in P$, $i_1 \in [i_3, i_4] \Leftrightarrow i_2 \in [i_3, i_4]$.
3. no nesting:
$\forall (i_1, i_2), (i_3, i_4) \in P$, $i_1 < i_3 + 1 \Leftrightarrow i_2 < i_3 + 1$.
4. no arcs:
$P = \emptyset$.

Following [3] these restrictions are used progressively and inclusively to produce five distinct levels of permitted arc structures for LAPCS:

– UNLIMITED — no restrictions;

(Advance online publication: 28 August 2012)

– CROSSING — restriction 1;
– NESTED — restrictions 1 and 2;
– CHAIN — restrictions 1, 2 and 3;
– PLAIN — restriction 4.

The problem LAPCS is varied by these different levels of restrictions as $\text{LAPCS}(x, y)$ which is problem LAPCS with S_1 having restriction level x and S_2 having restriction level y. Without loss of generality, we always assume that x is the same level or higher than y.

We give the definitions of two special cases of the LAPCS problem, which were first studied in [9]. The c-diagonal LAPCS problem is an extension of the c-fragment LAPCS problem, where base $S_2[i]$ is allowed only to match bases in the range $S_1[i - c, i + c]$.

In the decision version the c-fragment LAPCS problem $(c > 0)$ can be formulated as following.

INSTANCE: An alphabet Σ, annotated sequences S_1 and S_2, $S_1, S_2 \in \Sigma^*$, with arc sets P_1 and P_2 respectively, where S_1 and S_2 are divided into fragments of lengths exactly c (the last fragment can have a length less than c).

QUESTION: Is there a common subsequence T of S_1 and S_2 that is arc-preserving, $|T|+1 > k$? (The allowed matches are those between fragments at the same location).

The special cases are motivated from biological applications [14], [15]. The c-diagonal LAPCS and c-fragment LAPCS problems are relevant in the comparison of conserved RNA sequences where we already have a rough idea about the correspondence between bases in the two sequences.

The theory of parameterized computational complexity introduced in [16] is designed to address a natural and important qualitative complexity distinction which lies beyond **NP**-completeness.

A parameterized problem is a set $L \subseteq \Sigma^* \times \Sigma^*$ where Σ is a fixed alphabet. For convenience, we consider that a parameterized problem L is a subset of $L \subseteq \Sigma^* \times N$. We say that a parameterized problem L is uniformly fixed-parameter tractable if there is a constant α and an algorithm Φ such that Φ decides if $\langle x, k\rangle \in L$ in time $f(k)|x|^\alpha$ where $f : N \to N$ is an arbitrary function. If the function f is recursive, then L is strongly uniformly fixed-parameter tractable.

A problem L reduces to L' by a uniform parameterized reduction if there is an algorithm Φ which transforms $\langle x, k\rangle$ into $\langle x', g(k)\rangle$ in time $f(k)|x|^\alpha$, where $f, g : N \to N$ are arbitrary functions, and α is a constant independent of k, and $\langle x, k\rangle \in L$ if and only if $\langle x', g(k)\rangle \in L'$. As before, if f is recursive then the reduction is termed a strong uniform parameterized reduction.

Notably, most reductions from classical complexity turn out not to be parameterized ones [17]. For instance, the well-known reduction from INDEPENDENT SET to VERTEX COVER (see [18]) is not a parameterized one. This is due to the fact that the reduction function of the parameter $g(k)$ strongly depends on the instance itself, hence contradicting the definition of a parameterized reduction. However, the reductions from INDEPENDENT SET to CLIQUE and vice versa, which are obtained by simply passing the original graph over to the complementary one for $g(k) = k$, indeed are parameterized ones. Therefore, these problems are of comparable difficulty in terms of parameterized complexity.

The classes of the W hierarchy are based intuitively on the complexity of the circuits required to check solutions. A Boolean circuit defined to be of mixed type if it consists of circuits having gates of the following kinds:

- Small gates: *not* gates, *and* gates, and *or* gates with bounded fan-in.
- Large gates: *and* gates and *or* gates with unrestricted fan-in.

The depth of a circuit C is defined to be the maximum number of gates (small or large) on an input-output path in C. The weft of a circuit C is the maximum number of large gates on an input-output path in C.

We say that a family of decision circuits F has bounded depth if there is a constant h such that every circuit in the family F has depth at most h. We say that F has bounded weft if there is a constant t such that every circuit in the family F has weft at most t. The weight of a boolean vector x is the number of 1's in the vector.

Let F be a family of decision circuits. We allow that F may have many different circuits with a given number of inputs. To F we associate the parameterized circuit problem $L_F = \{\langle C, k\rangle : C \text{ accepts an input vector of weight } k\}$.

A parameterized problem L belongs to $W[t]$ if L reduces to the parameterized circuit problem $L_{F(t,h)}$ of mixed type decision circuits of weft at most t, and depth at most h, for some constant h. A parameterized problem L belongs to $W[P]$ if L reduces to the circuit problem L_F, where F is the set of all circuits. We designate the class of fixed-parameter tractable problems FPT.

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq W[P].$$

A problem which lets all other problems in $W[i]$ reduce to it is called $W[i]$-hard; if, additionally, it is contained in $W[i]$ then it is called $W[i]$-complete. See [17] for further discussion.

The arc structure can provide many natural parameters for LAPCS. For example, we can consider following three examples of such parameters concerning arc structure. Given an arc-annotated sequence (S, P), the cutwidth of the arc structure is the maximum number of arcs that pass by or end at any position of the sequence. Given an arc-annotated sequence (S, P), the bandwidth of the arc structure is the maximum distance between the two endpoints of an arc. The degree of an arc-annotated sequence (S, P) with unlimited arc structure is the maximum number of arcs from P that start or end in a base in S.

III. PREVIOUS RESULTS

It is shown in [9] that the problem c-diagonal LAPCS (nested, nested) (c-fragment LAPCS (nested, nested)) admits a **PTAS**. The 1-fragment LAPCS (crossing, crossing) and 0-diagonal LAPCS (crossing, crossing) are solvable in time $O(n)$ [9]. An overview on known **NP**-completeness results for c-diagonal LAPCS and c-fragment LAPCS is given in Table I.

Since many of variations of LAPCS are **NP**-hard or have no currently known polynomial time algorithm, the parameterized complexity of these problems has also been investigated. The parameters being used include: the length l of the desired subsequence, the cutwidth k of the arc

TABLE I
NP-COMPLETENESS RESULTS [9] FOR c-DIAGONAL LAPCS (WITH $c > 0$) AND c-FRAGMENT LAPCS (WITH $c > 1$)

	unlimited	crossing	nested	chain	plain
unlimited	**NP**	**NP**	**NP**	?	?
crossing	—	**NP**	**NP**	?	?
nested	—	—	**NP**	?	?

structure, bandwidth d of the arc structure, the degree b of an arc-annotated sequence with unlimited arc-structure.

If parameterized by the length of the desired subsequence, the LAPCS problem with at least one sequence having an unlimited arc structure was shown to be $W[1]$-complete [3]. If the arc structures of both sequences are crossing, the problem also turns out to be $W[1]$-complete [3]. It is shown in [3] that there exists an algorithm whose running time $O(9^k nm)$, where k is the cutwidth or bandwidth of the arc structure, to solve the LAPCS problem for variations with arc structure of both sequences being at most crossing. If the arc structures of both sequences are at most nested, then the LAPCS problem, parameterized by the cutwidth or bandwidth k of the arc structure, is fixed-parameter tractable and can be solved in time $O(k^2 4^k nm)$ [3]. The c-fragment LAPCS (crossing, crossing) problem, parameterized by the length l of the desired subsequence, is fixed-parameter tractable and can be solved in time $O((B+1)^l B^2 + c^3 n)$, where $B = c^2 + 2c - 1$ (see [7]). The c-diagonal LAPCS (crossing, crossing) problem, parameterized by the length l of the desired subsequence, is fixed-parameter tractable and can be solved in time $O((B+1)^l B^2 + c^3 n)$, where $B = 2c^2 + 7c + 2$ (see [7]).

Note that the c-fragment LAPCS (unlimited, unlimited) problem and the c-diagonal LAPCS (unlimited, unlimited) problem are fixed-parameter tractable, when the parameters are the length l of the desired subsequence and the maximum degree b of the two sequences: they can be solved in time $O((B+1)^l B^2 + (c^3 + 2bc^2)n)$, where $B = c^2 + 2bc - 1$, and in time $O((B'+1)^l B'^2 + (c^3 + 2bc^2)n)$, where $B' = 2c^2 + (4b+3)c + 2b$, respectively.

IV. $W[1]$-COMPLETENESS

Let us consider a parameterized version of the c-fragment LAPCS problem, parameterized by the length l of the desired subsequence.

c-fragment LAPCS[l]:

INSTANCE: An alphabet Σ, a positive integer l, annotated sequences S_1 and S_2, $S_1, S_2 \in \Sigma^*$, with arc sets P_1 and P_2 respectively, where S_1 and S_2 are divided into fragments of lengths exactly c (the last fragment can have a length less than c).

PARAMETER: l.

QUESTION: Is there a common subsequence T of S_1 and S_2 that is arc-preserving, $|T|+1 > l$? (The allowed matches are those between fragments at the same location).

Theorem 1. *If $|\Sigma| = 1$, then 1-fragment LAPCS[l] (unlimited, plain) is $W[1]$-complete.*

Proof. Let $G = (V, E)$ be an undirected graph, and let $I \subseteq V$. We say that the set I is independent if whenever $i, j \in I$ then there is no edge between i and j.

Let us consider the k-INDEPENDENT SET problem.

INSTANCE: A graph $G = (V, E)$.

PARAMETER; k.

QUESTION: Is there an independent set I, $I \subseteq V$, with $|I| + 1 > k$?

Note that the INDEPENDENT SET is **NP**-complete (see [18]) and the k-INDEPENDENT SET problem is $W[1]$-complete (see [17], [19]).

Let us suppose that $\Sigma = \{a\}$. We will show that the k-INDEPENDENT SET problem can be strong uniform parameterized reduced to the 1-fragment LAPCS[l] (unlimited, plain) problem.

Let $G = (V, E)$ where $V = \{1, 2, \ldots, n\}$. Let $\langle x, k\rangle$ be an instance of k-INDEPENDENT SET where $x = (V, E)$. Now we transform an instance $\langle x, k\rangle$ of the k-INDEPENDENT SET problem to an instance $\langle x', g(k)\rangle$ of the 1-fragment LAPCS[l] (unlimited, plain) problem as follows.

- $S_1 = S_2 = a^n$.
- $P_1 = E, P_2 = \emptyset$.
- $x' = (S_1, P_1), (S_2, P_2)$.
- $g(k) = k$.

It is easy to see that there is an algorithm Φ which transforms $\langle x, k\rangle$ into $\langle x', g(k)\rangle$ in time $f(k)|x|^\alpha$ where $\alpha = 1$ and $f(k) = 2$ is a recursive function.

Suppose that the graph G has an independent set I of size k. By definition of independent set, $(i, j) \notin E$ for each $i, j \in I$. For a given subset I, let $M = \{(i, i) : i \in I\}$. Since I is an independent set, if $(i, j) \in E = P_1$ then either $(i, i) \notin M$ or $(j, j) \notin M$. This preserves arcs since P_2 is empty. Clearly, $S_1[i] = S_2[i]$ for each $i \in I$, and the allowed matches are those between fragments at the same location. Therefore, there is a common subsequence T of S_1 and S_2 that is arc-preserving, $|T| = k$, and the allowed matches are those between fragments at the same location.

Now suppose that there is a common subsequence T of S_1 and S_2 that is arc-preserving, $|T| = k$, and the allowed matches are those between fragments at the same location. In this case there is a valid mapping M, with $|M| = k$. Since $c = 1$, it is easy to see that if $(i, j) \in M$ then $i = j$. Let $I = \{i : (i, i) \in M\}$. Clearly, $|I| = |M| = k$. Let i_1 and i_2 be any two distinct members of I. Then let $(i_1, j_1), (i_2, j_2) \in M$. Since $i_1 = j_1, i_2 = j_2, i_1 \neq i_2$, it is easy to see that $j_1 \neq j_2$. Since P_2 is empty, it is clear that $(j_1, j_2) \notin P_2$. So, $(i_1, i_2) \notin P_1$. Since $P_1 = E$, the set I of vertices is a size k independent set of G.

It is easy to see that our reduction is a strong uniform parameterized reduction. Therefore, the k-INDEPENDENT SET problem can be strong uniform parameterized reduced to 1-fragment LAPCS[l] (unlimited, plain). Since the l-INDEPENDENT SET problem is $W[1]$-complete the 1-fragment LAPCS[l] (unlimited, plain) is $W[1]$-hard.

Now we will show that the 1-fragment LAPCS[l] (unlimited, plain) problem can be strong uniform parameterized reduced to the k-INDEPENDENT SET problem.

Let $S_1 = a^n$, $S_2 = a^m$. If $m > n$, then it is easy to see that there is a common subsequence T of S_1 and S_2 that is arc-preserving, $|T| + 1 > l$, and the allowed matches are those between fragments at the same location if and only if there is a common subsequence T' of S_1 and a^n that is arc-preserving, $|T'| + 1 > l$, and the allowed matches are those between fragments at the same location. Therefore, we can suppose that $n + 1 > m$. Since the allowed matches are those between fragments at the same location, if $n > m$, then for arc-annotated sequences (S_1, P) and $(S_2, \emptyset)$, $S_1 = a^n$, $S_1 = a^m$, and $c = 1$ there is a common subsequence T of a^n and a^m that is arc-preserving, $|T| + 1 > l$, and the allowed matches are those between fragments at the same location if and only if for arc-annotated sequences $(S_1[1, n], Q)$ and $(S_2, \emptyset)$ there is a common subsequence T' of $S_1[1, n]$ and a^n that is arc-preserving, $|T'| + 1 > l$, and the allowed matches are those between fragments at the same location where $Q = \{(i_1, i_2) \mid (i_1, i_2) \in P, i_2 < n + 1\}$. Therefore, we can suppose that $n = m$.

Let $\langle x, l\rangle$ be an instance of the 1-fragment LAPCS[l] (unlimited, plain) problem where $x = (S_1, P_1), (S_2, P_2)$. Now we transform an instance $\langle x, l\rangle$ of the 1-fragment LAPCS[l] (unlimited, plain) problem to an instance $\langle x', g(l)\rangle$ of the k-INDEPENDENT SET problem as follows.

- $V = S_1$.
- $E = P_1$.
- $x' = (V, E)$.
- $g(l) = l$.

It is easy to see that there is an algorithm Φ which transforms $\langle x, l\rangle$ into $\langle x', g(l)\rangle$ in time $f(l)|x|^\alpha$ where $\alpha = 1$ and $f(l) = 2$ is a recursive function. Clearly, there is an independent set I, $I \subseteq V$, with $|I| + 1 > g(l)$, if and only if for arc-annotated sequences (S_1, P_1) and (S_2, P_2) and $c = 1$ there is a common subsequence T of S_1 and S_2 that is arc-preserving, $|T| + 1 > l$, and the allowed matches are those between fragments at the same location. Therefore, the 1-fragment LAPCS[l] (unlimited, plain) problem is $W[1]$-complete. □

V. LOGICAL MODELS OF c-FRAGMENT LAPCS (UNLIMITED, UNLIMITED)

The satisfiability problem (SAT) was the first known **NP**-complete problem. The problem SAT is the problem of determining if the variables of a given boolean function in conjunctive normal form (CNF) can be assigned in such a way as to make the formula evaluate to true. Considered also different variants of SAT. The problem SAT remains **NP**-complete even if all expressions are written in conjunctive normal form with 3 variables per clause (3-CNF). The problem 3SAT is the problem of determining if the variables of a given 3-CNF can be assigned in such a way as to make the formula evaluate to true.

In practice, the satisfiability problem is fundamental in solving many problems in automated reasoning, computer-aided design, computer-aided manufacturing, machine vision, database, robotics, integrated circuit design, computer architecture design, and computer network design. In recent years, many optimization methods, parallel algorithms, and practical techniques have been developed for solving the satisfiability problem (see e.g. [20]).

It is natural to use a reduction to different variants of the satisfiability problem to solve computational hard problems. Encoding problems as Boolean satisfiability and solving them with very efficient satisfiability algorithms has recently caused considerable interest. There are several ways of SAT-encoding constraint satisfaction, clique, planning, coloring, the Hamiltonian cycle, and some other problems (see e.g. [21]–[28]). In this paper we consider reductions from c-fragment LAPCS (unlimited, unlimited) to SAT and 3SAT.

Consider an alphabet Σ, annotated sequences S_1 and S_2, $S_1, S_2 \in \Sigma^*$, with arc sets P_1 and P_2 respectively, where S_1 and S_2 are divided into fragments of lengths exactly c (the last fragment can have a length less than c). Let $|S_1| = u$, $|S_2| = v$,

$$\varphi_1 = \bigwedge_{\substack{1 \le i \le u,\\ 1 \le j[1] < j[2] \le k}} (\neg x[i, j[1]] \vee \neg x[i, j[2]]),$$

$$\varphi_2 = \bigwedge_{1 \le j \le k} \bigvee_{1 \le i \le u} x[i, j],$$

$$\varphi_3 = \bigwedge_{\substack{1 \le i[1] < i[2] \le u,\\ 1 \le j \le k}} (\neg x[i[1], j] \vee \neg x[i[2], j]),$$

$$\varphi_4 = \bigwedge_{\substack{1 \le i \le u,\\ 1 \le j \le k,\\ 1 \le s \le u,\\ 1 \le t \le k,\\ s > i,\\ t < j}} (\neg x[i, j] \vee \neg x[s, t]),$$

$$\psi_1 = \bigwedge_{\substack{1 \le i \le v,\\ 1 \le j[1] < j[2] \le k}} (\neg y[i, j[1]] \vee \neg y[i, j[2]]),$$

$$\psi_2 = \bigwedge_{1 \le j \le k} \bigvee_{1 \le i \le v} y[i, j],$$

$$\psi_3 = \bigwedge_{\substack{1 \le i[1] < i[2] \le v,\\ 1 \le j \le k}} (\neg y[i[1], j] \vee \neg y[i[2], j]),$$

$$\psi_4 = \bigwedge_{\substack{1 \le i \le v,\\ 1 \le j \le k,\\ 1 \le s \le v,\\ 1 \le t \le k,\\ s > i,\\ t < j}} (\neg y[i, j] \vee \neg y[s, t]),$$

$$\rho_1 = \bigwedge_{\substack{1 \le i[1] \le u,\\ 1 \le i[2] \le v,\\ 1 \le j \le k,\\ S_1[i[1]] \ne S_2[i[2]]}} (\neg x[i[1], j] \vee \neg y[i[2], j]),$$

$$\rho_2 = \bigwedge_{\substack{1 \le i[1] < i[2] \le u,\\ 1 \le i[3] < i[4] \le v,\\ 1 \le j[1] < j[2] \le k,\\ (i[1], i[2]) \in P_1,\\ (i[3], i[4]) \notin P_2}} (\neg x[i[1], j[1]] \vee$$

$$\neg x[i[2], j[2]] \vee \neg y[i[3], j[1]] \vee \neg y[i[4], j[2]]),$$

$$\rho_3 = \bigwedge_{\substack{1 \le i[1] < i[2] \le u, \\ 1 \le i[3] < i[4] \le v, \\ 1 \le j[1] < j[2] \le k, \\ (i[1], i[2]) \notin P_1, \\ (i[3], i[4]) \in P_2}} (\neg x[i[1], j[1]] \vee$$

$$\neg x[i[2], j[2]] \vee \neg y[i[3], j[1]] \vee \neg y[i[4], j[2]]),$$

$$\eta_1 = \bigwedge_{\substack{1 \le i[1] \le u, \\ 1 \le j \le k, \\ i[1] = ac + b, \\ 1 \le b \le c}} (\neg x[i[1], j] \vee$$

$$(\bigvee_{ac+1 \le i[2] \le a(c+1)} y[i[2], j])),$$

$$\eta_2 = \bigwedge_{\substack{1 \le i[2] \le v, \\ 1 \le j \le k, \\ i[2] = ac + b, \\ 1 \le b \le c}} (\neg y[i[2], j] \vee$$

$$(\bigvee_{ac+1 \le i[1] \le a(c+1)} x[i[1], j])),$$

$$\xi = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \psi_4 \wedge$$

$$\rho_1 \wedge \rho_2 \wedge \rho_3 \wedge \eta_1 \wedge \eta_2.$$

Theorem 2. *Given an alphabet Σ, annotated sequences S_1 and S_2, $S_1, S_2 \in \Sigma^*$, with arc sets P_1 and P_2 respectively, where S_1 and S_2 are divided into fragments of lengths exactly c. The last fragment can have a length less than c. There is a common subsequence T of S_1 and S_2 that is arc-preserving, $|T| + 1 > k$, and the allowed matches are those between fragments at the same location if and only if ξ is satisfiable.*

Proof. Suppose that $\xi = 1$. Since $\varphi_1 = 1$, it is easy to see that, for all i, there is no more than one value of j such that $x[i, j] = 1$. Similarly, from $\varphi_3 = 1$ we obtain that, for all j, there is no more than one value of i such that $x[i, j] = 1$. In view of $\varphi_2 = 1$, it is clear that, for all j, there is at least one value of i such that $x[i, j] = 1$. Therefore, for all j, there is only one value of i such that $x[i, j] = 1$. Thus, we can consider values of $x[i, j]$ as a choice of elements of S_1. In particular, we can suppose that if $x[i, j] = 1$, then $S_1[i] \to T[j]$. Similarly, in view of $\psi_1 = \psi_2 = \psi_3 = 1$, we can consider values of $y[i, j]$ as a choice of elements of S_2. Note that at this time we are talking only about the existence of the mapping but not on its properties.

From $\varphi_4 = \psi_4 = 1$ we obtain that the choice defined by $x[i, j]$ and $y[i, j]$ is the order-preserving. In view of $\rho_1 = 1$, S_1 and S_2 are mapped into the same subsequence T. Since $\rho_2 = \rho_3 = 1$, it is easy to check that T is arc-preserving. In view of $\eta_1 = \eta_2 = 1$, it is clear that the allowed matches are those between fragments at the same location.

Now suppose that for annotated sequences S_1 and S_2 there is a common subsequence T of S_1 and S_2 that is arc-preserving, $|T| + 1 > k$, and the allowed matches are those between fragments at the same location. Without loss of generality, we can assume that $|T| = k$. Let $x[i, j] = 1$ if and only if $S_1[i] \to T[j]$. Let $y[i, j] = 1$ if and only if $S_2[i] \to T[j]$. It can be verified directly that in case of such values of variables $\xi = 1$. □

Clearly, ξ is a CNF. So, ξ gives us an explicit reduction from c-fragment LAPCS (unlimited, unlimited) to SAT.

Note that

$$\begin{aligned}
\alpha \quad &\Leftrightarrow \quad (\alpha \vee \beta_1 \vee \beta_2) \wedge \\
&\quad (\alpha \vee \neg\beta_1 \vee \beta_2) \wedge \\
&\quad (\alpha \vee \beta_1 \vee \neg\beta_2) \wedge \\
&\quad (\alpha \vee \neg\beta_1 \vee \neg\beta_2), && (1) \\
\vee_{j=1}^{l} \alpha_j \quad &\Leftrightarrow \quad (\alpha_1 \vee \alpha_2 \vee \beta_1) \wedge \\
&\quad (\wedge_{i=1}^{l-4} (\neg\beta_i \vee \alpha_{i+2} \vee \beta_{i+1})) \wedge \\
&\quad (\neg\beta_{l-3} \vee \alpha_{l-1} \vee \alpha_l), && (2) \\
\alpha_1 \vee \alpha_2 \quad &\Leftrightarrow \quad (\alpha_1 \vee \alpha_2 \vee \beta) \wedge \\
&\quad (\alpha_1 \vee \alpha_2 \vee \neg\beta), && (3) \\
\vee_{j=1}^{4} \alpha_j \quad &\Leftrightarrow \quad (\alpha_1 \vee \alpha_2 \vee \beta_1) \wedge \\
&\quad (\neg\beta_1 \vee \alpha_3 \vee \alpha_4) && (4)
\end{aligned}$$

where $l > 4$. Using relations (1) – (4) we can obtain an explicit transformation of ξ into τ such that $\xi \Leftrightarrow \tau$ and τ is a 3-CNF. Clearly, τ gives us an explicit reduction from c-fragment LAPCS (unlimited, unlimited) to 3SAT.

VI. A MODEL OF STATE EVOLUTIONS CAUSED BY ACTIONS

Reasoning about actions is a vital aspect for intelligent robots. It represents a major research domain in artificial intelligence. We can mention planning problems [21], [25], [28], [30], pattern recognition [31]–[33], pattern matching [23], [27], [34]–[38], localization problems [22], [26], [30], [39], mapping problems [24], [40], [41], self-awareness [42]–[45], etc. Reasoning about actions relies on the ability of relating cause and effect. In particular, we can consider state evolutions caused by actions (e.g. [46], [47]).

A state is a complete description of a situation the system can be in. Actions cause state transitions, making the system evolve from the current state to the next one. In principle we could represent the behavior of a system (i.e. all its possible evolutions) as a transition graph, where:

- Each node represents a state, and is labeled with the properties that characterize the state.
- Each arc represents a state transition, and is labeled by the action that causes the transition.

Note that complete knowledge of the behavior of the system is required to build its transition graph. Even if we had access to complete knowledge of the behavior of the system and can build its transition graph, we would still not succeed in creating a good model.

For a movement system with many degrees of freedom, there is an exponential explosion in the number of actions that can be taken in every state. In general one has only partial knowledge of such behavior. Even if we have a way to get the full knowledge about the robotic system, efforts to obtain such knowledge may be too high (e.g.

[48]). That is why in practice it is usually considered only a partial transition graph. Consideration of alternative models represents a significant interest.

Functioning of the domestic robot usually consists of performing some repetitive sequences of actions that are necessary for solving some tasks. Performing a sequence of actions a robot passes through a sequence of states. We can consider such sequence of states as a sequence $S \in \Sigma^*$ where Σ is the alphabet of states of the robot. Performed by a robot general task can be divided into local tasks that may be of interest for other general tasks. Without loss of generality, we can assume that each local task requires the same number of state transitions. In particular, we can use the dummy state. Respectively, S can be divided into fragments of lengths exactly c. Actions which are not included in the sequence performed by the robot but can it implemented are determine the arc annotation of S. In particular, we have considered the working day for vacuum cleaning robot. Selected experimental results are shown in Table II.

TABLE II
THE AVERAGE TIME (IN MINUTES) OF CLEANING FOR DIFFERENT ENVIRONMENTS.

The size of environment (m^2)	10	50	100	200
With arc annotation	9	42	79	137
Without arc annotation	9	46	98	209

Note that the case $c = 1$ is of special interest. In practice, it can be used for planning of large complex problems. For example, planning of the working day of a robot. When comparing two such plans, we are more interested in the study of arc structures rather than replacing letters.

Finding common subsequences of two such sequences is of interest for several robotic problems.

- We can use common subsequences for a more reasonable division into local tasks.
- Finding common subsequences allows to speed up training and calibration.
- We can optimize memory and simplify hardware implementation.

VII. Mining for Interesting Patterns

Feature selection is one of the most important problems of image processing (see e.g. [49], [50]). Note that a common technique used is the discovery of patterns which are frequent and happen often. The model of c-fragment LAPCS (unlimited, unlimited) can be used for mining for interesting patterns. We can use fluents [51] to express temporal patterns. A fluent is a proposition with temporal extent. In particular, a fluent can be represented as a finite binary time series $x[t]$. Using of fluents allows us to establish a correspondence between sequences of images and events. Recognition system creates a sequence

$$\mathrm{Im}[t[1]], \mathrm{Im}[t[2]], \ldots, \mathrm{Im}[t[n]], \ldots$$

of recognized and classified images where

$$t[1], t[2], \ldots, t[n], \ldots$$

are time points at which images were obtained. We can consider some event x. We can assume that $x[t]$ is 1 if and only if event x recognized on image $\mathrm{Im}[t]$. Now we can consider an arc-annotated sequence $A = (\mathrm{Im}, P)$ such that

$$(t[i], t[j]) \in P$$

if and only if

$$x[t[i-1]] = 0,$$

$$x[t[j+1]] = 0,$$

$$t[i] < t[j],$$

$$x[t[k]] = 1, i \le k \le j.$$

Note that the well-known problem of the longest common subsequence is a classical distance measure for strings. In particular, different versions of the longest common subsequence problem frequently used to mine interesting patterns (see e.g. [52]–[54]). In this case, we can mine interesting patterns using longest arc-preserving common subsequence technique. The model of c-fragment LAPCS allows us to take into account the time of the event.

Mining for interesting patterns has a number of interesting applications in robot self-awareness (see e.g. [42]–[45]). There are a number of different approaches to creating artificial emotions systems and systems of emotion recognition (see e.g. [55]). In particular, we need some system of prediction of collisions to build robot with ability to anticipate the motions (see e.g. [43], [56], [57]). We have considered models of c-fragment LAPCS and LCS over the set for mining for interesting patterns. We have used same data for c-fragment LAPCS and LCS over the set. We have mined two sets of interesting patterns. These sets were used by recurrent neural network for prediction of collisions of mobile robot. Selected experimental results are shown in Table III.

TABLE III
THE QUALITY OF PREDICTION FOR c-FRAGMENT LAPCS AND LCS OVER THE SET.

The size of training set	10^2	10^3	10^4	10^5
c-fragment LAPCS	91 %	96 %	97 %	98 %
LCS over the set	76 %	83 %	88 %	96 %

VIII. Experimental results

In the section V we have obtained explicit reductions from c-fragment LAPCS (unlimited, unlimited) to some variants of satisfiability, SAT and 3SAT. There is a well known site on which solvers for SAT are posted [29]. In addition to the solvers the site also represented a large set of test problems. This set includes a randomly generated problems of 3SAT. We have designed two generators of natural instances for c-fragment LAPCS (unlimited, unlimited). One of these generators creates instances on the basis of biological sequences. Since c-fragment LAPCS (unlimited, unlimited) is unnatural for biological data, we have considered biological sequences only for (nested, nested). Another generator creates instances on the basis of robotic information. For a domestic robot we have considered a model of state evolutions caused by actions. Also, we have considered a model of mining for interesting patterns.

For robotic test we consider sequences satisfying the following conditions. The model of state evolutions caused by actions: $400 \leq |S| \leq 600$; $1000 \leq |P| \leq 20000$. The model of mining for interesting patterns: $3000 \leq |S| \leq 12000$; $100 \leq |P| \leq 2000$. Also, we used random test problems from [29]. We used the algorithms fgrasp and posit from [29]. Also we design our own genetic algorithm (OA) for SAT which based on algorithms from [29]. We used heterogeneous cluster based on three clusters (Cluster USU, umt, um64) [58]. Each test was runned on a cluster of at least 100 nodes.

Selected experimental results for 3SAT are given in Tables IV – VII. Note that due to restrictions on computation time (20 hours) we used savepoints.

TABLE IV
EXPERIMENTAL RESULTS FOR 3SAT (RANDOM DATA)

time	fgrasp	posit	OA
average	6 h	5.8 h	6.3 h
max	168 h	174 h	84 h
best	19 min	26 min	3.6 min

TABLE V
EXPERIMENTAL RESULTS FOR 3SAT (BIOLOGICAL DATA)

time	fgrasp	posit	OA
average	36 min	28 min	6.4 min
max	87 h	93 h	19 h
best	11 min	17 min	56 sec

TABLE VI
EXPERIMENTAL RESULTS FOR 3SAT (THE MODEL OF STATE EVOLUTIONS CAUSED BY ACTIONS)

time	fgrasp	posit	OA
average	39 min	47 min	29.2 min
max	83 h	88 h	12 h
best	16 min	23 min	19 sec

TABLE VII
EXPERIMENTAL RESULTS FOR 3SAT (THE MODEL OF MINING FOR INTERESTING PATTERNS)

time	fgrasp	posit	OA
average	15 min	11 min	53 sec
max	17 h	19 h	2.6 h
best	5.6 min	4.2 min	7 sec

IX. CONCLUSION

In this paper, we have considered two special cases of the longest arc-preserving common subsequence problem, c-fragment LAPCS (unlimited, plain), c-fragment LAPCS (unlimited, unlimited). In particular, we have considered a parameterized version of the 1-fragment LAPCS (unlimited, plain) problem, parameterized by the length l of the desired subsequence. We have shown $W[1]$-completeness of the problem. Also, we have described an approach to solve c-fragment LAPCS (unlimited, unlimited). This approach is based on constructing logical models for the problem. We have considered different applications of the problem and experimental results for those applications.

REFERENCES

[1] V. Bafna, S. Muthukrishnan, and R. Ravi, "Comparing similarity between rna strings," in *Proceedings of the 6th Annual Symposium on Combinatorial Pattern Matching, 1995*, pp. 1-16.

[2] F. Corpet and B. Michot, "Rnalign program: alignment of rna sequences using both primary and secondary structures," *Computer Applications in the Biosciences*, vol. 10, no. 4, pp. 389-399, July 1994.

[3] P. A. Evans, *Algorithms and complexity for annotated sequence analysis*. Ph.D. thesis, University of Victoria, 1999.

[4] P. A. Evans, "Finding common subsequences with arcs and pseudoknots," in *Proceedings of the 10th Annual Symposium on Combinatorial Pattern Matching, 1999*, pp. 270-280.

[5] T. Jiang, G. H. Lin, B. Ma, and K. Zhang, "The longest common subsequence problem for arc-annotated sequences," in *Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching, 2000*, pp. 154-165.

[6] G. Blin, M. Crochemore, and S. Vialette, "Algorithmic Aspects of ARC-Annotated Sequences," in *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications, 2011*, pp. 171-183.

[7] J. Guo, *Exact Algorithms for the Longest Common Subsequence Problem for Arc-Annotated Sequences*. M.Sc. thesis, Eberhard-Karls-Universität, 2002.

[8] T. Jiang, G. H. Lin, B. Ma, and K. Zhang, "The longest common subsequence problem for arc-annotated sequences," *Journal of Discrete Algorithms*, vol. 2, no. 2, pp. 257-270, June 2004.

[9] G. H. Lin, Z. Z. Chen, T. Jiang, and J. J. Wen, "The longest common subsequence problem for sequences with nested arc annotations," *Journal of Computer and System Sciences*, vol. 65, no. 3, pp. 465-480, November 2002.

[10] J. Alber, J. Gramm, J. Guo, and R. Niedermeier, "Computing of two sequences with nested arc notations," *Theoretical Computer Science*, vol. 312, no. 2-3, pp. 337-358, January 2004.

[11] J. Gramm, J. Guo, and R. Niedermeier, "Pattern Matching fot arc-annotated sequences," *ACM Transactions on Algorithms*, vol. 2, no. 1, pp. 44-65, January 2006.

[12] D. Marx and I. Schlotter, "Parameterized complexity of the Arc-Preserving Subsequence problem," in *Proceedings of the 36th International Workshop on Graph-Theoretic Concepts in Computer Science, 2010*, pp. 244-255.

[13] A. Ouangraoua, V. Guignon, S. Hamel, and C. Chauve, "A new algorithm for aligning nested arc-annotated sequences under arbitrary weight schemes," *Theoretical Computer Science*, vol. 412, no. 8-10, pp. 753-764, March 2011.

[14] D. Gusfield, *Algorithm on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Cambridge, 1997.

[15] M. Li, B. Ma, and L. Wang, "Near optimal multiple alignment within a band in polynomial time," in *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, 2000*, pp. 425-434.

[16] R. Downey and M. Fellows, "Fixed-parameter intractability," in *Proceedings of the Seventh Annual Conference on Structure in Complexity Theory, 1992*, pp. 36-49.

[17] R. Downey and M. Fellows, *Parameterized complexity*. Springer-Verlag, New York, 1999.

[18] C. H. Papadimitriou, *Computational complexity*. Addison Wesley, Reading, 1994.

[19] R. Downey and M. Fellows, "Fixed-parameter tractability and completeness II: On completeness for $W[1]$," *Theoretical Computer Science*, vol. 141, no. 1-2, pp. 109-131, April 1995.

[20] J. Gu, P. Purdom, J. Franco, and B. Wah, "Algorithms for the Satisfiability (SAT) Problem: A Survey," in *Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge, 1996*, pp. 19-152.

[21] A. Gorbenko, M. Mornev, and V. Popov, "Planning a Typical Working Day for Indoor Service Robots," *IAENG International Journal of Computer Science*, vol. 38, no. 3, pp. 176-182, August 2011.

[22] A. Gorbenko, M. Mornev, V. Popov, and A. Sheka, "The problem of sensor placement for triangulation-based localisation," *International Journal of Automation and Control*, vol. 5, no. 3, pp. 245-253, August 2011.

[23] A. Gorbenko and V. Popov, "The set of parameterized k-covers problem," *Theoretical Computer Science*, vol. 423, no. 1, pp. 19-24, March 2012.

[24] A. Gorbenko and V. Popov, "On the Problem of Placement of Visual Landmarks," *Applied Mathematical Sciences*, vol. 6, no. 14, pp. 689-696, January 2012.

[25] A. Gorbenko and V. Popov, "Programming for Modular Reconfigurable Robots," *Programming and Computer Software*, vol. 38, no. 1, pp. 13-23, January 2012.

[26] A. Gorbenko, V. Popov, and A. Sheka, “Localization on Discrete Grid Graphs,” in *Proceedings of the CICA 2011, 2012*, pp. 971-978.
[27] A. Gorbenko and V. Popov, “The Longest Common Parameterized Subsequence Problem,” *Applied Mathematical Sciences*, vol. 6, no. 58, pp. 2851-2855, March 2012.
[28] A. Gorbenko and V. Popov, “On the Optimal Reconfiguration Planning for Modular Self-Reconfigurable DNA Nanomechanical Robots,” *Advanced Studies in Biology*, vol. 4, no. 2, pp. 95-101, March 2012.
[29] SATLIB — The Satisfiability Library. [Online]. Available: http://people.cs.ubc.ca/∼hoos/SATLIB/index-ubc.html
[30] A. Gorbenko and V. Popov, “The Binary Paint Shop Problem,” *Applied Mathematical Sciences*, vol. 6, no. 95, pp. 4733-4735, October 2012.
[31] A. Gorbenko and V. Popov, “Self-Learning Algorithm for Visual Recognition and Object Categorization for Autonomous Mobile Robots,” in *Proceedings of the CICA 2011, 2012*, pp. 1289-1295.
[32] A. Gorbenko and V. Popov, “On Face Detection from Compressed Video Streams,” *Applied Mathematical Sciences*, vol. 6, no. 96, pp. 4763-4766, October 2012.
[33] A. Gorbenko and V. Popov, “Usage of the Laplace Transform as a Basic Algorithm of Railroad Tracks Recognition,” *International Journal of Mathematical Analysis*, vol. 6, no. 48, pp. 2413-2417, October 2012.
[34] V. Popov, “The approximate period problem for DNA alphabet,” *Theoretical Computer Science*, vol. 304, no. 1-3, pp. 443-447, July 2003.
[35] V. Popov, “The Approximate Period Problem,” *IAENG International Journal of Computer Science*, vol. 36, no. 4, IJCS_36_4_03, November 2009.
[36] V. Popov, “Multiple genome rearrangement by swaps and by element duplications,” *Theoretical Computer Science*, vol. 385, no. 1-3, pp. 115-126, October 2007.
[37] A. Gorbenko and V. Popov, “The Longest Common Subsequence Problem,” *Advanced Studies in Biology*, vol. 4, no. 8, pp. 373-380, June 2012.
[38] A. Gorbenko and V. Popov, “Element Duplication Centre Problem and Railroad Tracks Recognition,” *Advanced Studies in Biology*, vol. 4, no. 8, pp. 381-384, June 2012.
[39] A. Gorbenko, M. Mornev, V. Popov, and A. Sheka, “The Problem of Sensor Placement,” *Advanced Studies in Theoretical Physics*, vol. 6, no. 20, pp. 965-967, July 2012.
[40] A. Gorbenko and V. Popov, “The Problem of Selection of a Minimal Set of Visual Landmarks,” *Applied Mathematical Sciences*, vol. 6, no. 95, pp. 4729-4732, October 2012.
[41] A. Gorbenko and V. Popov, “A Real-World Experiments Setup for Investigations of the Problem of Visual Landmarks Selection for Mobile Robots,” *Applied Mathematical Sciences*, vol. 6, no. 96, pp. 4767-4771, October 2012.
[42] A. Gorbenko and V. Popov, “Robot Self-Awareness: Occam’s Razor for Fluents,” *International Journal of Mathematical Analysis*, vol. 6, no. 30, pp. 1453-1455, March 2012.
[43] A. Gorbenko and V. Popov, “The Force Law Design of Artificial Physics Optimization for Robot Anticipation of Motion,” *Advanced Studies in Theoretical Physics*, vol. 6, no. 13, pp. 625-628, March 2012.
[44] A. Gorbenko, V. Popov, and A. Sheka, “Robot Self-Awareness: Exploration of Internal States,” *Applied Mathematical Sciences*, vol. 6, no. 14, pp. 675-688, January 2012.
[45] A. Gorbenko, V. Popov, and A. Sheka, “Robot Self-Awareness: Temporal Relation Based Data Mining,” *Engineering Letters*, vol. 19, no. 3, pp. 169-178, August 2011.
[46] W. Jacak, “Strategies of searching for collision-free manipulator motions: automata theory approach,” *Robotica*, vol. 7, no. 2, pp. 129-138, March 1989.
[47] F. Kanehiro, M. Inaba, and H. Inoue, “StateNet: State Transition Graph Description of Action Space that Includes Error Recovery Function,” *Journal of the Robotics Society of Japan*, vol. 20, no. 8, pp. 835-843, August 2002.
[48] A. Gorbenko, A. Lutov, M. Mornev, and V. Popov, “Algebras of Stepping Motor Programs,” *Applied Mathematical Sciences*, vol. 5, no. 34, pp. 1679-1692, June 2011.
[49] Y. Lin, M. D. McCool, and A. A. Ghorbani, “Time Series Motif Discovery and Anomaly Detection Based on Subseries Join,” *IAENG International Journal of Computer Science*, vol. 37, no. 3, IJCS_37_3_08, August 2010.
[50] P. Lin, N. Thapa, I. S. Omer, L. Liu, and J. Zhang, “Feature Selection: A Preprocess for Data Perturbation,” *IAENG International Journal of Computer Science*, vol. 38, no. 2, pp. 168-175, May 2011.
[51] T. Oates and P. R. Cohen, “Learning planning operators with conditional and probabilistic effects,” in *Proceedings of the AAAI Spring Symposium on Planning with Incomplete Information for Robot Problems, 1996*, pp. 86-94.
[52] W.-H. Hsu, Y.-Y. Chiang, and J.-S. Wu, “IntegratingWeighted LCS and SVM for 3D Handwriting Recognition on Handheld Devices using Accelerometers,” *WSEAS Transactions on Computers*, vol. 9, no. 3, pp. 235-251, March 2010.
[53] C. H. Park, J. W. Yoo, and A. M. Howard, “Transfer of skills between human operators through haptic training with robot coordination,” in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation, 2010*, pp. 229-235.
[54] P. Rashidi, D. J. Cook, L. B. Holder, and M. Schmitter-Edgecombe, “Discovering Activities to Recognize and Track in a Smart Environment,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 4, pp. 527-539, April 2011.
[55] S. Sumpeno, M. Hariadi, and M. H. Purnomo, “Facial Emotional Expressions of Life-like Character Based on Text Classifier and Fuzzy Logic,” *IAENG International Journal of Computer Science*, vol. 38, no. 2, pp. 122-133, May 2011.
[56] T. Kobayashi and S. Tsuda, “Sliding Mode Control of Space Robot for Unknown Target Capturing,” *Engineering Letters*, vol. 19, no. 2, pp. 105-111, May 2011.
[57] O. A. Dahunsi and J. O. Pedro, “Neural Network-Based Identification and Approximate Predictive Control of a Servo-Hydraulic Vehicle Suspension System,” *Engineering Letters*, vol. 18, no. 4, EL_18_4_05, November 2010.
[58] Web page “Computational resources of IMM UB RAS”. (In Russian.) [Online]. Available: http://parallel.imm.uran.ru/mvc_now/hardware/supercomp.htm

Anna Gorbenko was born on December 28, 1987. She received her M.Sc. in Computer Science from Department of Mathematics and Mechanics of Ural State University in 2011. She is currently a graduate student at Mathematics and Computer Science Institute of Ural Federal University and a researcher at Department of Intelligent Systems and Robotics of Mathematics and Computer Science Institute of Ural Federal University. She has (co-)authored 2 books and 18 papers. She has received Microsoft Best Paper Award from international conference in 2011.

Vladimir Popov was born on December 15, 1969. He received his Diploma in Mathematics (=M.Sci.) from Department of Mathematics and Mechanics of Ural State University in 1992. He was awarded his Candidate of Physical and Mathematical Sciences (=PhD) degree from Mathematics and Mechanics Institute of Ural Branch of Russian Academy of Sciences in 1996. He was awarded his Doctor of Physical and Mathematical Sciences degree from Mathematics and Mechanics Institute of Ural Branch of Russian Academy of Sciences in 2002. He is currently the chair of Department of Intelligent Systems and Robotics of Mathematics and Computer Science Institute of Ural Federal University and a Professor at Department of Mathematics and Mechanics of Mathematics and Computer Science Institute of Ural Federal University. He has (co-)authored 18 books and more than 120 papers. He has received Microsoft Best Paper Award from international conference in 2011. In 2008 his paper won the Russian competitive selection of survey and analytical papers.