Planning a Typical Working Day for Indoor Service Robots

Anna Gorbenko, Maxim Mornev, Vladimir Popov

Abstract—**This paper describes a method of considering the problem of planning for indoor service robot. We introduce a novel graph theoretic formulation of the planning a typical working day problem. We prove that this problem is NP-complete. In proof we consider an explicit reduction of the planning a typical working day problem to the problem of finding a Hamiltonian path. Also we describe an approach to solve this problem. This approach is based on constructing a logical model for the planning a typical working day problem. Using our reduction from the planning a typical working day problem to the Hamiltonian path problem and reduction from the Hamiltonian path problem to the satisfiability problem one can try to solve the planning a typical working day problem by local search and intelligent algorithms which developed for the satisfiability problem.**

Index Terms—**theory of actions, planning algorithms, satisfiability problem, Hamiltonian path problem.**

I. Introduction

PROBLEMS of planning and scheduling are among the most rapidly developing areas of modern computer science (see e.g. [1] – [5]). Different planning problems for mobile vehicles are of considerable interest for many years. For example, the vehicle routing problem was first introduced in 1959 [6] but still actual. The vehicle routing problem and its variants have been intensively studied and received considerable attention for many decades (see e.g. [7], [8] and references). Recently, in [9] presented a new distribution and route planning problem, general delivery problem. Among other examples we can mention multi-robot forest coverage problems (e.g. [10]), localization problems (e.g. [11], [12]), allocating complex tasks problems (e.g. [13]), path and motion planning problems (e.g. [14] – [18]), pursuit-evasion problems (e.g. [19]). In this paper we consider a planning problem for service robots. In particular, for such robots we consider a planning problem from the point of view of well-known and intensively studied theory of actions (e.g. [20] – [23]).

Service robots have no strict internationally accepted definition, which, among other things, delimits them from other types of equipment, in particular, the manipulating industrial robot. The ISRA (International Service Robot Association — One of the several Robotic Industries Association (RIA) Organizations) [24] defines service robots as machines that sense, think, and act to benefit or extend human capabilities and to increase human productivity. According to the IFR (International Federation of Robotics) [25], a service robot is a robot which operates semi- or fully autonomously to perform services useful to the well-being of humans and equipment, excluding manufacturing operations. With this definition, manipulating industrial robots could also be regarded as service robots, provided they are installed in non-manufacturing operations. Service robots may or may not be equipped with an arm structure as is the industrial robot. Often, but not always, the service robots are mobile. In some cases, service robots consist of a mobile platform on which one or several arms are attached and controlled in the same mode as the arms of the industrial robot.

For a long time, the use of robots was limited to manufacturing automation and space exploration. Only large corporations and government agencies could afford the systems integration costs associated with the robotics automation projects. Underdeveloped artificial intelligence technology, safety issues and high costs of systems integration were the main limiting factors preventing the use of robotics by small businesses and households. Service robots have long been a staple of science fiction and commercial visions of the future. Until recently, we have only been able to speculate about what the experience of using such a device might be. Current service robots, introduced as consumer products, allow us to make this vision a reality. Though the service robots are in their infancy stage, we are witnessing perhaps the most exciting and promising robot evolution of all. Service robots will one day become the largest class of robot applications, outnumbering the industrial uses by several times.

Because they are designed for a social world, service robots must carry out functional as well as non-functional (social) tasks. In the future, autonomous, mobile service robots will assist people in many environments. Robots could help the elderly and caretakers, assist with work around the home, act as guards, and perform tasks that are repetitive, boring, or dangerous in nursing homes, hospitals, military environments, disaster sites, and schools.

Current service robot investigations dial with following robot specializations [24]: agriculture and harvesting robots; automatic refilling robots; cleaning and housekeeping robots; construction robots; edutainment robots; fire fighting robots; robots for domestic tasks; robots in food industry; guides and office robots; humanitarian demining robots; humanoid and anthropomorphic robots; inspection robots; lawn mowing robots; medical robotics; mining robots; picking and palletizing robots; rehabilitation robots; surveillance and exploration robots; search and rescue robots. In this paper we consider multi-task indoor service robots such as robots for domestic tasks and office robots. For such robot we can suppose that during a typical working day it must perform some fixed set of tasks. It is natural to consider for such robots the planning a typical working day problem which require to create a proper sequence of tasks. We introduce a novel graph theoretic formulation of this problem and prove that it is **NP**-complete. Also we describe an approach to solve this

Ural State University, Department of Mathematics and Mechanics, 620083 Ekaterinburg, Russian Federation. Email: gorbenko.aa@gmail.com, max.mornev@gmail.com, Vladimir.Popov@usu.ru

The work was partially supported by Analytical Departmental Program "Developing the scientific potential of high school" 2.1.1/14055.

problem.

II. The planning a typical working day for indoor service robots problem

Consider an indoor service robot R. Suppose that S is the set of everyday jobs of R.

Following a common approach in reasoning about actions, dynamic systems are modeled in terms of state evolutions caused by actions. 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 $G = (V, E)$,

$$V = \{v_1, v_2, \dots, v_n\},$$
$$E = \{e_1, e_2, \dots, e_m\},$$

where:

- Each node $v_i \in V$ represents a state, and is labeled with the properties that characterize the state.
- Each arc $e_j \in E$ represents a state transition, and is labeled by the action that causes the transition.

Since we consider a working day for robot, we can suppose that there exist an initial state $\mathcal{S}$. Correspondingly, we can assume that there exist a final state $\mathcal{F}$. So, during a typical working day a service robot R performs some path from $\mathcal{S}$ to $\mathcal{F}$.

For each task $s \in S$ suppose that during a typical working day task s can be solved by R only $t(s)$ times where

$$t(s) \in \{1, 2, \dots\} \cup \{\infty\}.$$

For example, let R be a cooking robot and

$$S = \{breakfast, dinner, supper, beep, water\}.$$

It is natural to assume that

$$t(breakfast) = 1,$$
$$t(dinner) = 1,$$
$$t(supper) = 1,$$
$$t(beep) = \infty,$$
$$t(water) > 1, t(water) \neq \infty.$$

In general case a task $s \in S$ require a sequence of transitions

$$v_{i_1} \overset{e_{j_1}}{\to} v_{i_2} \overset{e_{j_2}}{\to} \cdots \overset{e_{j_{k-1}}}{\to} v_{i_k}.$$

We suppose that during a working day a task s solved by our robot $t(s)$ times if and only if during this day it performs some path from $\mathcal{S}$ to $\mathcal{F}$ such that node v_{i_k} visited $t(s)$ times. Therefore, we can assume that $S \subseteq V$.

For some service robots we also needed a limitation for nodes from $V \backslash S$. For example, water cleaning robots can move along carpets but we may want to limit with activity. Therefore, for each node $v \in V \backslash S$ we suppose that during a typical working day node v can be visited by R only $f(v)$ times where

$$f(v) \in \{1, 2, \dots\} \cup \{\infty\}.$$

From this point of view essential difference between nodes from S and from $V \backslash S$ consists in following conditions:

- Each node $s \in S$ must be visited.
- Each node $v \in V \backslash S$ can be visited.

Now, we can introduce a graph theoretic formulation of the planning a typical working day for indoor service robots problem. Consider the following problem:

Planning a typical working day for indoor service robots problem:

Instance: *A set S, a transition graph $G = (V, E)$, a function $t : S \to \{1, 2, \dots\} \cup \{\infty\}$.*

Question: *Does G have a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V \backslash S$ visited no more then $f(v)$ times?*

Theorem. *The planning a typical working day for indoor service robots problem is* **NP***-complete.*

Proof. Note that it is not evident that the planning a typical working day for indoor service robots problem is in **NP**, since in general case a solution of this problem may be not of polynomial size. To show that our problem is in **NP** we reduce the planning a typical working day for indoor service robots problem to the Hamiltonian path problem.

Lemma 1. *Let $t(v) = p$, $t(v) > 1$, $t(v) \neq \infty$, for some transition graph $G = (V, E)$ and set S, $v \in S$. Let*

$$S' = (S \backslash \{v\}) \cup \{v_{n+1}, v_{n+2}, \dots, v_{n+p}\}. \quad (1)$$

Let $G' = (V', E')$ be a graph such that

$$V' = \{v_1, v_2, \dots, v_{n+p}\} \backslash \{v\},$$
$$E' = \{e_1, e_2, \dots, e_m,$$
$$(v_i, v_{n+r})_l, (v_{n+r}, v_j)_l, (v_{n+d}, v_{n+d+1})_1 \mid (v_i, v)_l \in E,$$
$$(v, v_j)_l \in E, 1 \leq r \leq p, 1 \leq d \leq p-1\} \backslash$$
$$\{(v_i, v)_l, (v, v_j)_l \mid (v_i, v)_l \in E, (v, v_j)_l \in E\},$$
$$t(v_{n+r}) = 1, 1 \leq r \leq p.$$

In G there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V \backslash S$ visited no more then $f(v)$ times if and only if in G' there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V' \backslash S'$ visited no more then $f(v)$ times.

Proof of Lemma 1. Let a sequence

$$v_{i_1}, v_{i_2}, \dots, v_{i_k} \quad (2)$$

be a path in $G = (V, E)$ from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V \backslash S$ visited no more then $f(v)$ times. Consider a set

$$\{j[1], j[2], \dots, j[q]\}$$

such that

$$j \in \{j[1], j[2], \dots, j[q]\}$$

if and only if $v_{i_j} = v$. Suppose that

$$j[1] < j[2] < \cdots < j[q]. \tag{3}$$

By definition of a sequence (2),

$$q \le t(v) = p.$$

Therefore, by definition of G', we can consider in G' a sequence in which all occurrences of v replaced by different occurrences of nodes from

$$\{v_{n+1}, v_{n+2}, \dots, v_{n+p}\}.$$

In view of (3), we can consider the following sequence:

$$\begin{gathered} v_{i_1}, v_{i_2}, \dots, v_{i_{j[1]-1}}, v_{n+1}, \\ v_{i_{j[1]+1}}, v_{i_{j[1]+2}}, \dots, v_{i_{j[2]-1}}, v_{n+2}, \\ v_{i_{j[2]+1}}, v_{i_{j[2]+2}}, \dots, v_{i_{j[q]-1}}, v_{n+q}, \\ v_{i_{j[q]+1}}, v_{i_{j[q]+2}}, \dots, v_{i_k}. \end{gathered} \tag{4}$$

By definition of a sequence (2),

$$\begin{gathered} (v_{i_{j[1]-1}}, v)_{l[1]} \in E, (v, v_{i_{j[1]+1}})_{l[2]} \in E, \\ (v_{i_{j[2]-1}}, v)_{l[3]} \in E, (v, v_{i_{j[2]+1}})_{l[4]} \in E, \\ \dots \\ (v_{i_{j[q]-1}}, v)_{l[2q-1]} \in E, (v, v_{i_{j[q]+1}})_{l[2q]} \in E \end{gathered}$$

for some

$$l[1], l[2], \dots, l[2q].$$

Therefore, by definition of G',

$$\begin{gathered} (v_{i_{j[1]-1}}, v_{n+1})_{l[1]} \in E', (v_{n+1}, v_{i_{j[1]+1}})_{l[2]} \in E', \\ (v_{i_{j[2]-1}}, v_{n+2})_{l[3]} \in E', (v_{n+2}, v_{i_{j[2]+1}})_{l[4]} \in E', \\ \dots \\ (v_{i_{j[q]-1}}, v_{n+q})_{l[2q-1]} \in E', (v_{n+q}, v_{i_{j[q]+1}})_{l[2q]} \in E'. \end{gathered} \tag{5}$$

From (5) and definition of a sequence (2) we obtain that the sequence (4) is a path in the graph G'. By definition of G' and (4),

$$\begin{gathered} v_{i_1}, v_{i_2}, \dots, v_{i_{j[1]-1}}, v_{n+1}, \\ v_{i_{j[1]+1}}, v_{i_{j[1]+2}}, \dots, v_{i_{j[2]-1}}, v_{n+2}, \\ v_{i_{j[2]+1}}, v_{i_{j[2]+2}}, \dots, v_{i_{j[q]-1}}, v_{n+q}, \\ v_{n+q+1}, v_{n+q+2}, \dots, v_{n+p}, \\ v_{i_{j[q]+1}}, v_{i_{j[q]+2}}, \dots, v_{i_k}. \end{gathered} \tag{6}$$

is a path in the graph G'. Since (2) is a path in G from $\mathcal{S}$ to $\mathcal{F}$, (4) is a path in the G' from $\mathcal{S}$ to $\mathcal{F}$. Since (2) is a path in G such that robot R solves all tasks from S, in view of (1), (6) is a path in G' such that robot R solves all tasks from S'. Moreover, it is easy to see that (6) is a path in G' such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V' \backslash S'$ visited no more then $f(v)$ times.

Now, let a sequence (2) be a path in G' from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V' \backslash S'$ visited no more then $f(v)$ times. Consider a set

$$\{j[1], j[2], \dots, j[q]\}$$

such that

$$j \in \{j[1], j[2], \dots, j[q]\}$$

if and only if $i_j > n$. We can suppose that $j[l_1] > j[l_2]$ if and only if $l_1 > l_2$. Consider a sequence

$$\begin{gathered} v_{i_1}, v_{i_2}, \dots, v_{i_{j[1]-1}}, v, \\ v_{i_{j[1]+1}}, v_{i_{j[1]+2}}, \dots, v_{i_{j[2]-1}}, v, \\ v_{i_{j[2]+1}}, v_{i_{j[2]+2}}, \dots, v_{i_{j[q]-1}}, v, \\ v_{i_{j[q]+1}}, v_{i_{j[q]+2}}, \dots, v_{i_k}. \end{gathered} \tag{7}$$

By definition of a sequence (2),

$$\begin{gathered} (v_{i_{j[1]-1}}, v_{n+1})_{l[1]} \in E', (v_{n+1}, v_{i_{j[1]+1}})_{l[2]} \in E', \\ (v_{i_{j[2]-1}}, v_{n+2})_{l[3]} \in E', (v_{n+2}, v_{i_{j[2]+1}})_{l[4]} \in E', \\ \dots \\ (v_{i_{j[q]-1}}, v_{n+q})_{l[2q-1]} \in E', (v_{n+q}, v_{i_{j[q]+1}})_{l[2q]} \in E' \end{gathered}$$

for some

$$l[1], l[2], \dots, l[2q].$$

Therefore, by definition of G',

$$\begin{gathered} (v_{i_{j[1]-1}}, v)_{l[1]} \in E, (v, v_{i_{j[1]+1}})_{l[2]} \in E, \\ (v_{i_{j[2]-1}}, v)_{l[3]} \in E, (v, v_{i_{j[2]+1}})_{l[4]} \in E, \\ \dots \\ (v_{i_{j[q]-1}}, v)_{l[2q-1]} \in E, (v, v_{i_{j[q]+1}})_{l[2q]} \in E. \end{gathered}$$

From these relations and definition of a sequence (2) we obtain that the sequence (7) is a path in the graph G. Since

$$t(v_{n+r}) = 1, 1 \le r \le p,$$

it is easy to see that $q \le p$. Now, it is clear that (7) is a path in $G = (V, E)$ from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V \backslash S$ visited no more then $f(v)$ times.

So, in G there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V \backslash S$ visited no more then $f(v)$ times if and only if in G' there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V' \backslash S'$ visited no more then $f(v)$ times. $\square$

In view of Lemma 1, we can assume that $t(v) \in \{1, \infty\}$, $v \in S$.

Lemma 2. *Let* $t(v) = \infty$ *for some transition graph* $G = (V, E)$ *and set* S, $v \in S$. *Let*

$$S' = (S\backslash\{v\}) \cup \{v_{n+1}\}.$$

Let $G' = (V', E')$ be a graph such that

$$V' = \{v_1, v_2, \ldots, v_{n+2}\},$$

$$E' = \{e_1, e_2, \ldots, e_m,$$

$$(v_i, v_{n+r})_l, (v_{n+r}, v_j)_l, (v_{n+1}, v_{n+2})_1 \mid (v_i, v)_l \in E,$$

$$(v, v_j)_l \in E, 1 \le r \le 2\}\backslash$$

$$\{(v_i, v)_l, (v, v_j)_l \mid (v_i, v)_l \in E, (v, v_j)_l \in E\},$$

$$t(v_{n+1}) = 1,$$

$$f(v_{n+2}) = \infty.$$

In G there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V\backslash S$ visited no more then $f(v)$ times if and only if in G' there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V'\backslash S'$ visited no more then $f(v)$ times.

Proof of Lemma 2. Let a sequence

$$v_{i_1}, v_{i_2}, \ldots, v_{i_k} \tag{8}$$

be a path in $G = (V, E)$ from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V\backslash S$ visited no more then $f(v)$ times. Consider a set

$$\{j[1], j[2], \ldots, j[q]\}$$

such that

$$j \in \{j[1], j[2], \ldots, j[q]\}$$

if and only if $v_{i_j} = v$. Suppose that

$$j[1] < j[2] < \cdots < j[q]. \tag{9}$$

By definition of a sequence (8), $q \ge 1$. Therefore, in view of (9), we can consider the following sequence:

$$\begin{gathered} v_{i_1}, v_{i_2}, \ldots, v_{i_{j[1]-1}}, v_{n+1}, \\ v_{i_{j[1]+1}}, v_{i_{j[1]+2}}, \ldots, v_{i_{j[2]-1}}, v_{n+2}, \\ v_{i_{j[2]+1}}, v_{i_{j[2]+2}}, \ldots, v_{i_{j[q]-1}}, v_{n+2}, \\ v_{i_{j[q]+1}}, v_{i_{j[q]+2}}, \ldots, v_{i_k}. \end{gathered} \tag{10}$$

By definition of a sequence (8),

$$(v_{i_{j[1]-1}}, v)_{l[1]} \in E, (v, v_{i_{j[1]+1}})_{l[2]} \in E,$$

$$(v_{i_{j[2]-1}}, v)_{l[3]} \in E, (v, v_{i_{j[2]+1}})_{l[4]} \in E,$$

$$\ldots$$

$$(v_{i_{j[q]-1}}, v)_{l[2q-1]} \in E, (v, v_{i_{j[q]+1}})_{l[2q]} \in E$$

for some

$$l[1], l[2], \ldots, l[2q].$$

Therefore, by definition of G',

$$\begin{gathered} (v_{i_{j[1]-1}}, v_{n+1})_{l[1]} \in E', (v_{n+1}, v_{i_{j[1]+1}})_{l[2]} \in E', \\ (v_{i_{j[2]-1}}, v_{n+2})_{l[3]} \in E', (v_{n+2}, v_{i_{j[2]+1}})_{l[4]} \in E', \\ \ldots \\ (v_{i_{j[q]-1}}, v_{n+2})_{l[2q-1]} \in E', (v_{n+2}, v_{i_{j[q]+1}})_{l[2q]} \in E' \end{gathered} \tag{11}$$

From (11) and definition of a sequence (8) we obtain that the sequence (10) is a path in the graph G'.

Since (8) is a path in G from $\mathcal{S}$ to $\mathcal{F}$, (10) is a path in the G' from $\mathcal{S}$ to $\mathcal{F}$. Since (8) is a path in G such that robot R solves all tasks from S, in view of $S' = (S\backslash\{v\}) \cup \{v_{n+1}\}$, (10) is a path in G' such that robot R solves all tasks from S'. Moreover, it is easy to see that (10) is a path in G' such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V'\backslash S'$ visited no more then $f(v)$ times.

Now, let a sequence (8) be a path in G' from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V'\backslash S'$ visited no more then $f(v)$ times. It is clear, that in this case we can simply replace all occurrences of v_{n+1} and v_{n+2} by v. So, in G there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V\backslash S$ visited no more then $f(v)$ times if and only if in G' there exist a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S' and each node $v \in V'\backslash S'$ visited no more then $f(v)$ times. $\square$

In view of Lemma 2, we can assume that $t(v) = 1$, $v \in S$.

For some transition graph $G = (V, E)$ consider a graph $G' = (V, E')$ such that

$$E' = E\backslash\{(v, v)_p, (u, v)_l \mid l > 1, (v, v)_p \in E, (u, v)_l \in E\}.$$

It is easy to see that G have a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V\backslash S$ visited no more then $f(v)$ times if and only if G' have a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V\backslash S$ visited no more then $f(v)$ times. Therefore, we can assume that $(v, v)_p \notin E$, $(u, v)_l \notin E$ for any $u, v \in V$ and $l > 1$.

Let for some transition graph $G = (V, E)$ and set S a sequence

$$v_{i_1}, v_{i_2}, \ldots, v_{i_k}$$

be a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V\backslash S$ visited no more then $f(v)$ times. Assume that

$$v_{i_p} \in S, v_{i_{p+q}} \in S,$$

$$v_{i_r} \notin S, p+1 \le r \le p+q-1,$$

$$v_{i_a} = v_{i_b}, p+1 \le a < b \le p+q-1.$$

Then it is easy to see that a sequence

$$v_{i_1}, v_{i_2}, \ldots, v_{i_a}, v_{i_{b+1}}, \ldots, v_{i_k}$$

be a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V'\backslash S'$ visited no more then $f(v)$ times. Therefore, for any transition graph $G = (V, E)$, set S, and node $v \in V\backslash S$ we can suppose that during a typical working day node v can be visited by R only $|S|+1$ times. So, we can suppose that $f(v) \leq |S| + 1$, $v \in V\backslash S$.

Consider a graph $G' = (V', E')$ such that

$$V\backslash S = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\},$$

$$V' = S \cup \{a_{1,1}, a_{1,2}, \ldots, a_{1,g_1}\}\cup$$

$$\{a_{2,1}, a_{2,2}, \ldots, a_{2,g_2}\}\cup$$

$$\ldots$$

$$\{a_{k,1}, a_{k,2}, \ldots, a_{k,g_k}\}\cup$$

$$\{\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_g\},$$

$$g_p = f(v_{i_p}), 1 \leq p \leq k,$$

$$g = 2\sum_{p=1}^{k} g_p,$$

$$E' = (((E\backslash\{(e, \mathcal{F})_1 \mid e \in V\})\cup$$

$$\{(e, \mathcal{F}_0)_1 \mid (e, \mathcal{F})_1 \in E, e \in V\})\backslash$$

$$\{(e, v_{i_j})_1, (v_{i_j}, e)_1 \mid e \in V, 1 \leq j \leq k\})\cup$$

$$\{(e, a_{j,l})_1, (a_{j,l}, e)_1 \mid 1 \leq j \leq k, 1 \leq l \leq g_j\}\cup$$

$$\{(\mathcal{F}_i, \mathcal{F}_{i+1})_1 \mid 0 \leq i \leq g-1\}\cup$$

$$\{(\mathcal{F}_i, a_{j,l})_1, (a_{j,l}, \mathcal{F}_{i+1})_1 \mid$$

$$i = 2\sum_{p=1}^{j-1} g_p + 2l - 1,$$

$$1 \leq j \leq k, 1 \leq l \leq g_j\}\cup$$

$$\{(\mathcal{F}_g, \mathcal{F})_1\}.$$

Clearly, graph G' have a Hamiltonian path if and only if graph G have a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V'\backslash S'$ visited no more then $f(v)$ times.

It is easy to see that this transformation can be done in polynomial time. Therefore, we need to prove only **NP**-hardness of the planning a typical working day for indoor service robots problem. For this purpose we reduce the Hamiltonian path problem to our problem.

A Hamiltonian path in a graph G is a path that visits each graph node exactly once. Let us consider the following problem:

HAMILTONIAN PATH PROBLEM:

Instance: A directed graph $D = (A, B)$.

Question: Does D *have a Hamiltonian path?*

The Hamiltonian path problem is **NP**-complete (cf. [26]).

Now, we transform an instance of the Hamiltonian path problem into an instance of the planning a typical working day for indoor service robots problem as follows:

$$V = A \cup \{\mathcal{S}, \mathcal{F}\},$$

$$E = B \cup \{(\mathcal{S}, e), (e, \mathcal{F}) \mid e \in B\},$$

$$S = V,$$

$$t(s) = 1, s \in S.$$

It is easy to see that this transformation can be done in polynomial time and logarithmic space.

Suppose that G have a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S. Since $S = V$ and $t(s) = 1$, $s \in S$, this path in a graph G is a path that visits each graph node exactly once. By definition, this is a Hamiltonian path for G. Note that in graph G there are no arcs from $\mathcal{F}$ to $x \in V\backslash\{\mathcal{F}\}$. Correspondingly, in graph G there are no arcs from $x \in V\backslash\{\mathcal{S}\}$ to $\mathcal{S}$. Therefore, there exist a Hamiltonian path in the subgraph H of G generated by $V\backslash\{\mathcal{S}, \mathcal{F}\}$. It is easy to see that $H = D$.

Suppose now that

$$a_1, a_2, \ldots, a_k$$

is a Hamiltonian path in D. Since

$$\{(\mathcal{S}, e), (e, \mathcal{F}) \mid e \in B\} \subseteq E,$$

it is easy to see that

$$\mathcal{S}, a_1, a_2, \ldots, a_k, \mathcal{F}$$

is a Hamiltonian path from $\mathcal{S}$ to $\mathcal{F}$ in G. Since $t(s) = 1$, $s \in S$, by definition, this Hamiltonian path is a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S. Therefore, D have a Hamiltonian path if and only if G have a path from $\mathcal{S}$ to $\mathcal{F}$ such that robot R solves all tasks but no more then $t(s)$ times for any task s from set S and each node $v \in V'\backslash S'$ visited no more then $f(v)$ times. $\square$

III. A LOGICAL MODEL OF THE PLANNING A TYPICAL WORKING DAY FOR INDOOR SERVICE ROBOTS PROBLEM

The satisfiability problem is a core problem in mathematical logic and computing theory. In practice, SAT 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. Traditional methods treat SAT as a discrete, constrained decision problem. In recent years, many optimization methods, parallel algorithms, and practical techniques have been developed for solving SAT (see [27]).

In particular, proposed several genetic algorithms [28] – [31]. Considered hybrid algorithms in which the approach of genetic algorithms combined with local search [32]. Relatively high efficiency demonstrated by algorithms based solely on local search. Of course, these algorithms require exponential time at worst. But they can relatively quick receive solutions for many boolean functions. Therefore, it is natural to use a reduction to the SAT to solve computational hard problems.

Encoding problems as Boolean satisfiability and solving them with very efficient satisfiability algorithms has recently caused considerable interest. In particular, local search algorithms have given impressive results on many problems. For example, there are several ways of SAT-encoding constraint satisfaction (see e.g. [33] – [36]), clique [37], planning (see e.g. [38] – [40]), and coloring problems (see e.g. [37], [41]). There are a number of explicit reductions from the Hamiltonian cycle problem to SAT (see e.g. [37], [41], [42]).

In previous section we obtain an explicit reduction from the planning a typical working day for indoor service robots problem to the Hamiltonian path problem. Using simple combination of ideas from previous section and from [37] we obtain an explicit reduction from the planning a typical working day for indoor service robots problem to the satisfiability problem.

There is a well known site on which posted solvers for SAT [43]. Currently on the site published more then 10 implementations of algorithms for solving SAT. They are divided into two main classes: stochastic local search algorithms and algorithms improved exhaustive search. All solvers allow the conventional format for recording DIMACS Boolean function in conjunctive normal form and solve the corresponding problem [44]. In addition to the solvers the site also represented a large set of test problems in the format of DIMACS. This set includes randomly generated problems of 3SAT.

We create a generator of special hard and natural instances for the planning a typical working day for indoor service robots problem. Also we design our own genetic algorithm for SAT which based on algorithms from [43].

We use heterogeneous cluster based on three clusters (Cluster USU, Linux, 8 calculation nodes, Intel Pentium IV 2.40GHz processors; umt, Linux, 256 calculation nodes, Xeon 3.00GHz processors; um64, Linux, 124 calculation nodes, AMD Opteron 2.6GHz bi-processors) [45]. For computational experiment we create 161 special hard test sets and 206 natural test sets. Each test was run on a cluster of 100 nodes for 20 hours. For special hard test sets: the maximum solution time was 12 hours; the average time to find a solution was 21.5 minutes; the best time was 53 seconds. For natural test sets: the maximum solution time was 10 hours; the average time to find a solution was 5.5 minutes; the best time was 26 seconds.

IV. Conclusion

In this paper we consider the planning a typical working day for indoor service robots problem. We have proved that the problem is in **NP**. We considered an approach to solve this problem.

Note that many planning problems for mobile vehicles are **NP**-hard and therefore difficult to solve. Because getting the best results by applying the exact algorithm from instance computer program is very costly, therefore, intelligent algorithms are often desired. Often we can not characterize with assurance an intelligent algorithm that we use. Is this a really good algorithm? Maybe we just use too simple instances. We need hard instances to test our algorithms. Using hard instances requires finding a solver that could find the exact solutions for testing. If we have a solver for hard instances, we can create a good test bed for testing intelligent algorithms. In particular, we can use the explicit reduction obtained in this paper to create a test bed for the planning a typical working day for indoor service robots problem.

The problem is quite general. In particular, this problem can be used for planning for cleaning and housekeeping robots, robots for domestic tasks, guides and office robots, humanoid and anthropomorphic robots, inspection robots, picking and palletizing robots, surveillance and exploration robots, etc. Note that the considered problem is of interest not only for robots that have a set of typical repetitive tasks. Also, the problem is of interest for planning of individual repeating hard tasks. In particular, we can mention a visual calibration (see e.g. [46]). However, it is a plenty room for further investigations. In particular, in further research interesting to consider planning in dynamic systems and planning with a counteraction.

References

[1] P. Bertoli, M. Pistore, and P. Traverso, "Automated composition of Web services via planning in asynchronous domains," *Artificial Intelligence*, vol. 174, no. 3-4, pp. 316-361, March 2010.
[2] S. Srivastava, N. Immerman, and S. Zilberstein, "A new representation and associated algorithms for generalized planning," *Artificial Intelligence*, vol. 175, no. 2, pp. 615-647, February 2011.
[3] K. Wang and S. H. Choi, "Decomposition-Based Scheduling for Makespan Minimisation of Flexible Flow Shop with Stochastic Processing Times," *Engineering Letters*, vol. 18, no. 1, EL_18_1_09, February 2010.
[4] F. Wu, S. Zilberstein, and X. Chen, "Online planning for multi-agent systems with bounded communication," *Artificial Intelligence*, vol. 175, no. 2, pp. 487-511, February 2011.
[5] M. Zhongyi, M. Younus, and L. Yongjin, "Automated Planning and Scheduling System for the Composite Component Manufacturing Workshop," *Engineering Letters*, vol. 19, no. 1, pp. 75-83, February 2011.
[6] G. B. Dantzig and J. H. Ramser, "The Truck Dispatching Problem," *Management Science*, vol. 6, no. 1, pp. 80-91, October 1959.
[7] A. Boonkleaw, N. Suthikarnnarunai, and R. Srinon, "Strategic Planning for Newspaper Delivery Problem Using Vehicle Routing Algorithm with Time Window (VRPTW)," *Engineering Letters*, vol. 18, no. 2, EL_18_2_09, May 2010.
[8] G. Koloch and B. Kaminski, "Nested vs. Joint Optimization of Vehicle Routing Problems with Three-dimensional Loading Constraints," *Engineering Letters*, vol. 18, no. 2, EL_18_2_10, May 2010.
[9] L. Lian and E. Castelain, "A Decomposition Approach to Solve a General Delivery Problem," *Engineering Letters*, vol. 18, no. 1, EL_18_1_10, February 2010.
[10] X. Zheng, S. Koenig, D. Kempe, and S. Jain, "Multi-Robot Forest Coverage for Weighted and Unweighted Terrain," *IEEE Transactions on Robotics*, vol. 26, no. 6, pp. 1018-1031, November 2010.
[11] S. Koenig, J. Mitchell, A. Mudgal, and C. Tovey, "A Near-Tight Approximation Algorithm for the Robot Localization Problem," *SIAM Journal on Computing*, vol. 39, no. 2, pp. 461-490, March 2009.
[12] C. Tovey and S. Koenig, "Localization: Approximation and Performance Bounds to Minimize Travel Distance," *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 320-330, March 2010.
[13] X. Zheng and S. Koenig, "K-Swaps: Cooperative Negotiation for Solving Task-Allocation Problems," in *Proceedings of the 2009 International Joint Conference on Artificial Intelligence*, pp. 373-379.
[14] J.-W. Choi, R. E. Curry, and G. H. Elkaim, "Continuous Curvature Path Generation Based on Bezier Curves for Autonomous Vehicles," *IAENG International Journal of Applied Mathematics*, vol. 40, no. 2, IJAM_40_2_07, May 2010.
[15] K. Daniel, A. Nash, S. Koenig, and A. Felner, "Theta*: Any-Angle Path Planning on Grids," *Journal of Artificial Intelligence Research*, vol. 39, pp. 533-579, September 2010.
[16] A. Nash, S. Koenig, and M. Likhachev, "Incremental Phi*: Incremental Any-Angle Path Planning on Grids," in *Proceedings of the 2009 International Joint Conference on Artificial Intelligence*, 1824-1830.
[17] P. Sotiropoulos, N. Aspragathos, and F. Andritsos, "Optimum Docking of an Unmanned Underwater Vehicle for High Dexterity Manipulation," *IAENG International Journal of Computer Science*, vol. 38, no. 1, pp. 48-56, February 2011.
[18] W. Yue and J. Franco, "A New Way to Reduce Computing in Navigation Algorithm," *Engineering Letters*, vol. 18, no. 4, EL_18_4_03, November 2010.

[19] R. Borie, C. Tovey, and S. Koenig, "Algorithms and Complexity Results for Pursuit-Evasion Problems," in *Proceedings of the 2009 International Joint Conference on Artificial Intelligence*, pp. 59-66.
[20] M. Bienvenu, C. Fritz, S. A. McIlraith, "Specifying and computing preferred plans," *Artificial Intelligence*, vol. 175, no. 7-8, pp. 1308-1345, May 2011.
[21] M. Sridharan, J. Wyatt, R. Dearden, "Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs," *Artificial Intelligence*, vol. 174, no. 11, pp. 704-725, July 2010.
[22] M. Thielscher, "A unifying action calculus," *Artificial Intelligence*, vol. 175, no. 1, pp. 120-141, January 2011.
[23] P. H. Tu, T. C. Son, M. Gelfond, A. R. Morales, "Approximation of action theories and its application to conformant planning," *Artificial Intelligence*, vol. 175, no. 1, pp. 79-119, January 2011.
[24] Website of Robotic Industries Association — Robotics Online. [Online]. Available: http://www.robotics.org/
[25] Website of International Federation of Robotics. [Online]. Available: http://www.ifr.org/home/
[26] M. R. Garey and D. S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-completeness*. San Francisco, California, USA: W. H. Freeman and Co, 1979.
[27] J. Gu, P. Purdom, J. Franco, and B. Wah, "Algorithms for the Satisfiability (SAT) Problem: A Survey," in *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1996*, pp. 19-152.
[28] J. Fleurent, "Genetic algorithms and hybrids for graph coloring," *Annals of Operations Research*, vol. 63, no. 3, pp. 437-461, June 1996.
[29] J. Hao, R. Dorne, "A new population-based method for satisfiability problems," in *Proceedings of 11th European Conference on Artificial Intelligence, 1994*, pp. 135-139.
[30] K. Jong, W. Spears, "Using genetic algorithms to solve np-complete problems," in *Proceedings of the International Conference on Genetic Algorithms, 1989*, pp. 124-132.
[31] R. Voorn, M. Dastani, E. Marchiori, "Finding simplest pattern structures using genetic programming," in *Proceedings of the Genetic and Evolutionary Computation Conference, 2001*, pp. 3-10.
[32] J. Hao, F. Lardeux, F. Saubion, "A hybrid genetic algorithm for the satisfiability problem," in *Proceedings of the 1rst International Workshop on Heuristics, 2002*, pp. 102-109.
[33] C. Bessiere, E. Hebrard, and T. Walsh, "Local Consistencies in SAT," in *Theory and applications of satisfiability testing: SAT 2003: international conference on theory and applications of satisfiability testing N 6, 2003*, pp. 400-407.
[34] M. Davis, G. Logemann, and D. Loveland, "A Machine Program for Theorem Proving," *Communications of the ACM*, vol. 5, no. 7, pp. 394-397, July 1962.
[35] A. Frisch, and T. Peugniez, "Solving Non-Boolean Satisfiability Problems with Stochastic Local Search," in *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, 2001*, pp. 282-288.
[36] A. M. Frisch, T. J. Peugniez, A. J. Doggett, and P. W. Nightingale, "Solving Non-Boolean Satisfiability Problems with Stochastic Local Search: A Comparison of Encodings," *Journal of Automated Reasoning*, vol. 35, no. 1-3, pp. 143-179, March 2005.
[37] K. Iwama and S. Miyazaki, "SAR-variable complexity of hard combinatorial problems," in *Information Processing 94, IFIP Transactions A, Computer Science and Technology, vol. I*, pp. 253-258.
[38] M. Büttner, and J. Rintanen, "Improving parallel planning with constraints on the number of operators," in *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling, 2005*, pp. 292-299.
[39] M. Ernst, T. Millstein, and D. Weld, "Automatic SAT-Compilation of Planning Problems," in *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, 1997*, pp. 1169-1176.
[40] H. Kautz, "SATPLAN04: Planning as Satisfiability," in *Proceedings of the 4th International Planning Competition at the 14th International Conference on Automated Planning and Scheduling, 2004*, pp. 44-45.
[41] H. H. Hoos, "SAT-Encodings, Search Space Structure, and Local Search Performance," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence 1999*, pp. 296-302.
[42] A. D. Plotnikov, "A Logical Model of HCP," *International Journal of Mathematics and Mathematical Sciences*, vol. 26, no. 11, November 2001.
[43] SATLIB — The Satisfiability Library. [Online]. Available: http://people.cs.ubc.ca/∼hoos/SATLIB/index-ubc.html
[44] Satisfiability Suggested Format. [Online]. Available: http://www.cs.ubc.ca/∼hoos/SATLIB/Benchmarks/SAT/satformat.ps
[45] Web page "Computational resources of IMM UB RAS". (In Russian.) [Online]. Available: http://parallel.imm.uran.ru/mvc_now/hardware/supercomp.htm
[46] A. Gorbenko, A. Lutov, M. Mornev, V. Popov, "Algebras of Stepping Motor Programs," *Applied Mathematical Sciences*, vol. 5, no. 34, pp. 1679-1692, June 2011.

Anna Gorbenko was born on December 28, 1987. She received her B.Sc. in Computer Science from Department of Mathematics and Mechanics of Ural State University in 2009. Since 2009 she is a Researcher of the Department of Intelligent Systems and Robotics of Ural State University. She has (co-)authored 2 books and 17 papers. She has received Microsoft Best Paper Award from international conference in 2011.

Maxim Mornev was born on October 16, 1986. He received his B.Sc. in Computer Science from Department of Mathematics and Mechanics of Ural State University in 2009. Since 2006 he is a Researcher of the Department of Intelligent Systems and Robotics of Ural State University. He has (co-)authored 13 papers.

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 the Department of Intelligent Systems and Robotics of Ural State University and a Professor at Department of Mathematics and Mechanics of Ural State 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.