ICIC Express Letters
Part B: Applications
Volume 7, Number 1, January 2016

ICIC International ©2016 ISSN 2185-2766
pp. 77–83

MULTITASKING PLANNING FOR SERVICE ROBOT BASED ON HIERARCHICAL DECOMPOSING

Zhixian Chen[1,2,3], Jun Zhang[1,2,3,*], Ying Hu[1,2], Xuehui Dong[1]
Haiyang Jin[1] and Jianwei Zhang[4]

[1]Shenzhen Institutes of Advanced Technology
Chinese Academy of Sciences
No. 1068, Xueyuan Avenue, Shenzhen University Town, Shenzhen 518055, P. R. China
{ zx.chen; ying.hu; xh.dong; hy.jin }@siat.ac.cn; *Corresponding author: jun.zhang@siat.ac.cn

[2]The Chinese University of Hong Kong
Shatin, NT, Hong Kong SAR, P. R. China

[3]Shenzhen College of Advanced Technology
University of Chinese Academy of Sciences
No. 1068, Xueyuan Avenue, Shenzhen University Town, Shenzhen 518055, P. R. China

[4]TAMS
University of Hamburg
Mittelweg 177, Hamburg 20148, Germany

Received June 2015; accepted September 2015

Abstract. *In this paper, the proposed approach addresses the planning for multiple tasks in individual service robot and aims for planning them as a whole to increase the efficiency of their execution. To solve this multitasking planning problem, the model of HTN planning is expanded with the plan constituted with action cliques in which actions are compatible for executing in parallel. With two tree data structures for process data reusing, the decomposing algorithm is adapted to the extended model, and a re-planning algorithm based on partial backtracking is designed for plan repairmen. Furthermore, to parallel actions execution, an optimization method for action parallelizing is modeled as maximum weighted clique problem based on the rule of maximal cost saving first and evaluating the relations between the actions. Finally, the planner is developed and the experiments show its feasibility and improvement on the efficiency of time consumption for multitasking in individual robot both in rover domain in simulation and integrating in physical robot for delivery tasks.*

Keywords: Multitasking planning, Hierarchical decomposing, Maximum weight clique, Service robot

1. Introduction.

Today, more and more robots are developed for service in human's everyday life, e.g., ASIMO [1], and expected to be more flexible and efficient. As is known, robot planning has been integrated in many robot systems, and been proved to be an effective way to enhance robot's competence [2]. However, as McDermott [3] said, planning problem turned to be too hard and too easy. On one side, planning has been successfully practiced in motion planning for robots [4]. And on the higher level such as task planning, which always involves various operations, studies have shown the feasibility and practicability of robot planning on improving robot's competence for complex tasks. On the other side, for planning on multiple tasks, studies mostly dedicated their efforts in multi agents system [5] or simply used the series-wound planning strategy [6] which plans the tasks one by one and leads to very inefficient performances of tasks accomplishment. However, few of them focus on planning for multiple tasks problem and their execution in parallel in an individual robot. Even though in [7], the ideal to parallelize plan execution and re-planning on a mobile robot was presented, it is demonstrated by handwork prescription rather than automated planning.

To simplify description of this problem, we call it as *RM (Robot Multitasking)*, and a planning problem for it, *RMP (Robot Multitasking Planning)*, is to find a plan to increase the efficiency of executing multiple tasks for an individual robot. To solve an RMP, an approach, which plans the RM as a whole inspired by the nature cognition of human, and its planner system *PlanRM (Planner for Robot Multitasking)* are introduced.

2. **Approach and Formalism.** The main idea of our approach is to plan an RMP as a whole and to enable robot to perform multiple tasks in parallel. To realize this idea, the approach is divided into three phases: task decomposing, action parallelization and re-planning, as shown in Figure 1. In task decomposing, the HTN (Hierarchical Task Network) method [8], which has been well developed, is employed to generate and maintain the plan for each individual task. Finally, a task tree and a plan tree are generated to store the hierarchical task nets and plans for all tasks. In action parallelizing, an optimization based on maximal cost saving first is designed as the intercessor to select the actions to form an action clique from the winner task processes. At the meantime, the other task processes that are failure in optimization will be re-planned in re-planning phase. And the task tree and plan tree are trimmed after re-planning.

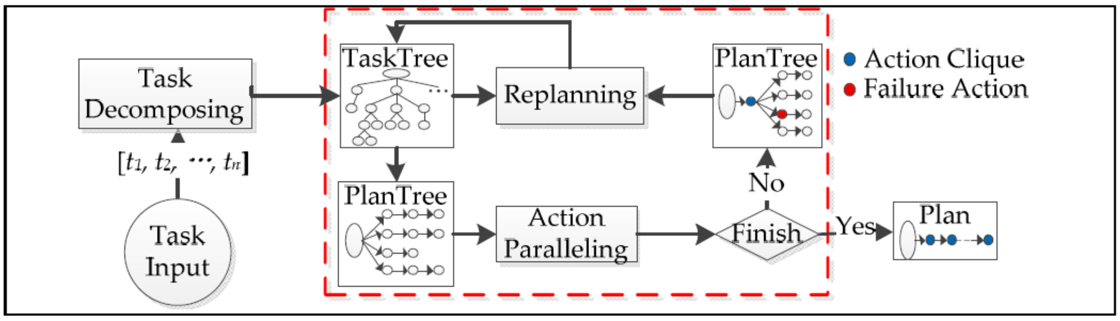


Figure 1. The approach for RMP

To formalize the RMP, the HTN formalism is adopted and expanded, which provides a syntax of defining the methods by controlling the decomposing of tasks and avoiding the irrelevant searching branches to speed up significantly the task decomposition [9]. The definitions in Table 1 are derived from the traditional HTN model and SHOP2 [8].

Table 1. Related definitions

Definition	Syntax	Explanation
State s	*(predicate, term, . . .),*	describes a fact of the world
Operator o	*(name (o), precond (o), add (o), delete (o))*	a generalization of primitive action
Method m	*(name (m), task (m), precond (m), subtask (m))*	a generalization of compound skill of robot
Planning domain	$D = (S, O, M)$	the limited sets respectively of states, operators and methods
Planning problem	$P = (D, S_0, T)$	the limited sets respectively of states, operators and methods
Action Clique AC	$\{a_1, a_2, \dots, a_i\}$	a set of actions a_i, in which actions are compatible with each other and able to perform in parallel
Plan	$\pi = (AC_1, \dots, AC_n)$	a total ordering sequence of action clique AC_i

3. **Task Decomposition and Re-planning.** As shown in Algorithm 1, the goal of task decomposing is to generate the task tree and the plan tree for non-primitive tasks. Each task is decomposed independently to get its hierarchical task net and plan by a function of decompose_task(). And then the TaskNode and PlanNode are attached respectively to

Algorithm 1: generating the task tree and plan tree based on hierarchical decomposing

```
generate_trees ($S_0$ , $Tasks$):
    $TaskTree$ ← Tree (name='$TaskTree$'), $PlanTree$ ← Tree (name='$PlanTree$')
    for $t_i$ ∈ $Tasks$:
        $TaskNode$ ← Tree (name=$t_i$), $PlanNode$ ← Tree (name=$t_i$)
        $TaskNode$, $PlanNode$ ← decompose_task ($S_0$, $t_i$ ,$TaskNode$, $PlanNode$)
        $TaskTree$.add_child ($TaskNode$), $PlanTree$.add_child ($PlanNode$)
decompose_task ($S_0$, $tasklist$, $tasktree$, $plantree$)
    if $tasklist$ is empty: return ($tasktree$, $plantree$)
    if $tasklist_0$ is an instance of operator:
        $S'$ ← $\gamma$ ($S_0$, $tasklist_0$)
        if $S'$ is not $False$: $plantree$.add_child ($tasklist_0$), $tasklist$.remove ($tasklist_0$)
            $tasktree$, $plantree$ ← decompose_task($S'$, $tasklist$, $tasktree$, $plantree$)
    if $tasklist_0$ is an instance of method:
        $subtasks$ ← method ($S_0$, $tasklist_0$).subtask, $tasklist$.remove ($tasklist_0$)
        $tasklist$ ← insert $subtasks$ in the front of $tasklist$
        $tasktree$.add_children ($subtasks$)
        $tasktree$, $plantree$ ← decompose_task ($S_0$, $tasklist$, $tasktree$, $plantree$)
    else: return $False$
```

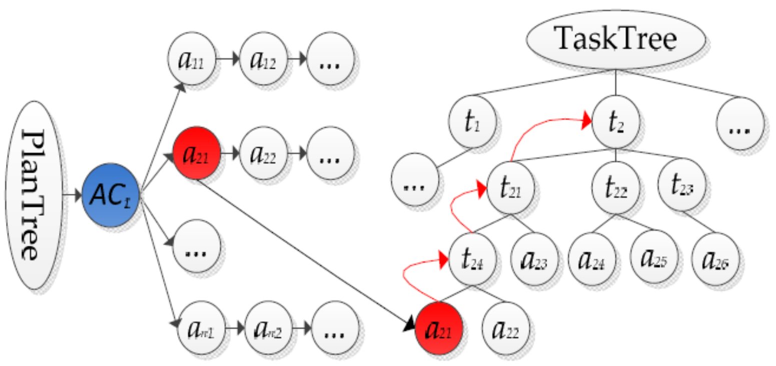


FIGURE 2. Basic principle of re-planning based on partial backtracking

TaskTree and PlanTree as sub-trees. Once the failure actions rise up, re-planningis called to repair them. A re-planning algorithm based on partial backtracking is developed for restricting the range of re-planning and improving the performance, as shown in Figure 2.

4. **Action Parallelization.**

To parallelize the whole plan, the action parallelization generates the action cliques by selecting the actions that are applicable for executing in parallel layer by layer top-down of the plan tree. Two requirements should be satisfied. One is that the actions in an action clique must be compatible with each other, and the other is that the final plan should be optimal one for multitasking.

Besides independence and retraction, another kind of relation for action parallelizing is defined as follows. Two actions a_i and a_j $(a_i \neq a_j)$ are with **allowance** relation, while:

$$[del(a_i) \cap pre(a_j)] \cup [add(a_i) \cap del(a_j)] = \emptyset$$

and

$$[pre(a_i) \cap del(a_j)] \cup [del(a_i) \cap add(a_j)] = \emptyset$$

The optimization is modelled as an MWCP (Maximum Weight Clique Problem) [10] based on the graph theory. The problem can be modelled as:

$$G = (A, W, E) \tag{1}$$

where A is a set of vertices representing action candidates for action clique; W is a set of weights for according actions; E is a set of edges connecting the relative actions.

An example graph of MWCP in rover domain is shown in Figure 3, where the vertices are named with actions and their weights are noted on the nodes. An edge indicates the relation between 2 connected actions, where an undirected edge presents an independence relation and a directed edge presents an allowance relation. For allowance relation, the arrow restricts that the output action must be executed before the input action.

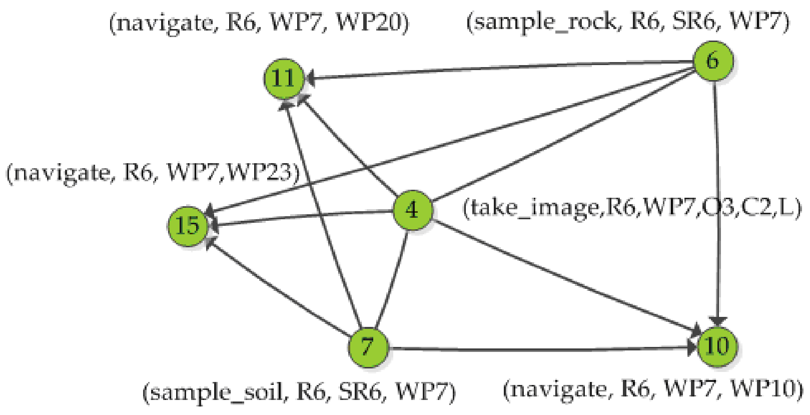

(navigate, R6, WP7, WP20)
(sample_rock, R6, SR6, WP7)
11
6
(navigate, R6, WP7,WP23)
15
4
(take_image,R6,WP7,O3,C2,L)
7
10
(sample_soil, R6, SR6, WP7)
(navigate, R6, WP7, WP10)

FIGURE 3. An example graph of MWCP in rover domain

The goal of optimization is to find an action clique with maximum of optimization function with the weights. It can be formalized as:

$$\max f(w_{a_i}, c_x),\ a_i \in c_x,\ c_x \text{ is a clique in } G \tag{2}$$

All cliques in graph G ignoring the weight should be found firstly, and the modular of $find\_cliques(G)$ [11] in Networkx [12] is employed. A set of cliques C can be got by:

$$C = \{c_x\} \leftarrow find\_cliques(G) \tag{3}$$

And the action clique AC is obtained as:

$$AC = \{c_x | f_{\max}(w_{a_i}, c_x)\} \tag{4}$$

The target of optimization in this paper is to reduce the time consumption for the multitasking of service robot. As an optimal action clique consists of allowance relations, the actions in clique have to be sorted by allowance order. For each clique $c_x \in C$, a set of action paths $AP(c_x)$ can be found by the modular of $find\_all\_simple\_paths(c_x)$ as (5).

$$AP(c_x) \leftarrow find\_all\_simple\_paths(c_x) \tag{5}$$

And in (6), the weight for each action path $ap_k \in AP(c_x)$ is counted respectively as the total weights of the actions in ap_k. As soon as the $W(ap_k)$ of all $AP(c_x)$ are got, the weight of clique c_x is the maximum of $W(ap_k)$s, shown in (7). Then we can get the cost saving of clique c_x with (8), where n_{a_i} is the number of a_i that appears in the active layer of plan tree. Finally, the target optimum can be obtained as (9).

$$W(ap_k) = \Sigma_{ap_k}(w_{a_i} | a_i \in ap_k) \tag{6}$$

$$W(c_k) = \max(W(ap_k) | ap_k \in AP(c_x)) \tag{7}$$

$$WS(c_x) = \Sigma_{c_x}(w_{a_i} * n_{a_i} | a_i \in c_x) - W(c_x) \tag{8}$$

$$f_{\max}(c_x, w_{a_i}) = \max(WS(c_x)) \tag{9}$$

5. **Experiment and Result.** To verify the feasibility and performance of our approach, the Mars Rover domain introduced in the 3rd international planning competition was used [14]. The task was to explore planetary to get the sample data of soil or rock from certain way points, or have images of some objects. The rover can only travel over certain terrain types and transmit the data to a lander. In the revised domain, there were 9 operators and 20 methods, and the problem was to solve 25 tasks of sampling soil data, sampling rock data and taking images in a topology map with 29 waypoints, shown as Figure 4. In addition, a cost dictionary was built at random to define the cost of every action with executing time. To compare our approach with the currents, the Pyhop which is a standard HTN planning system written as a Python library was chosen as the contrast. In the experiment, both planners were tested repeatedly for 10000 times respectively under the same setting to minimize the randomness of the result. Finally, both planners successfully found the appropriate plans in all tests. As shown in Table 2, our planner, PlanRM, had done an obvious progress to increase the efficiency of multitasking for the rover. On average, PlanRM reduced about 28.7% on the amount of actions and 37.4% on the cost for execution.

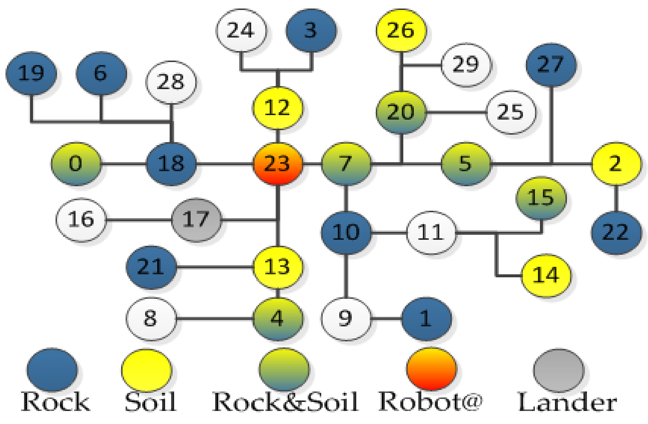
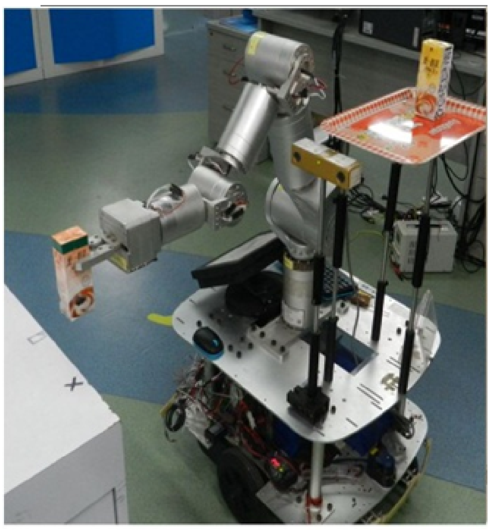


FIGURE 4. Topology map for rover domain

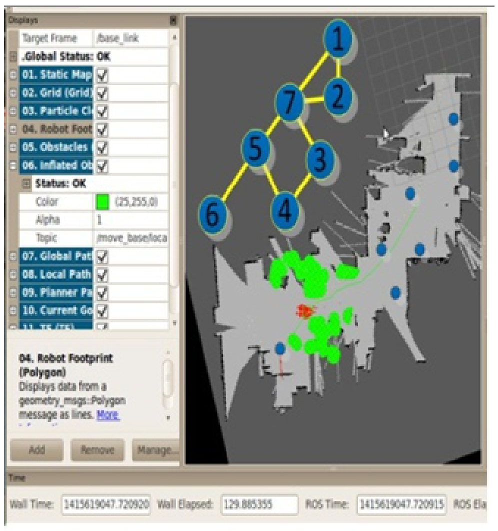
TABLE 2. Results for rover problems

Problem	Problem for ROVER6		
Planner	PlanRM	Pyhop	RD rate
Average AAC	179	251	28.7%
Average cost	978	1562	37.4%
Min AAC	145	212	31.6%
Max AAC	230	296	22.3%
Min cost	715	1270	43.7%
Max cost	1381	1897	27.2%

The experiment for physical robot is tested in a delivery service scenario on a mobile robot, which has ability of automated navigation and manipulation (shown as Figure 5). The environment for experiment is set with 7 tables distributed in a laboratory about 150m^2 with a terrain map. The tasks for robot are to deliver the drink among the tables. The same as in the experiment of rover domain, Pyhop is employed again as the competitor with PlanRM. An executor distributes the plan generated by planner to each action control modules and monitors their execution through the feedback. In the testing on the robot, as the accomplishment for six tasks will cost so much time, we reduce the number of the tasks to three. The time cost for every action execution is recorded and the result is shown



(a) Mobile robot with a 7-DOF arm



(b) Working map in RVIZ simulator

FIGURE 5. The robot in delivery domain

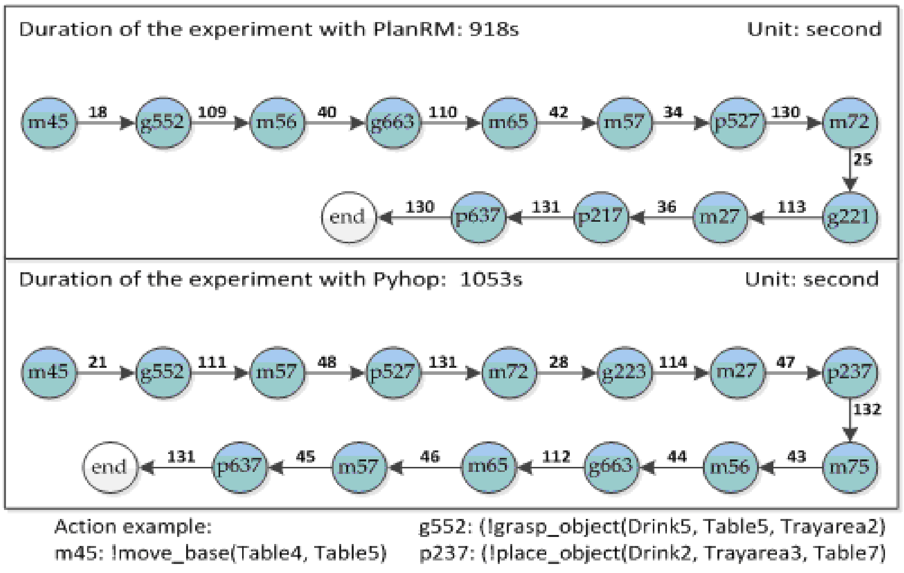

Duration of the experiment with PlanRM: 918s
Unit: second
m45 18 g552 109 m56 40 g663 110 m65 42 m57 34 p527 130 m72 25 g221 113 m27 36 p217 131 p637 130 end
Duration of the experiment with Pyhop: 1053s
Unit: second
m45 21 g552 111 m57 48 p527 131 m72 28 g223 114 m27 47 p237 132 m75 43 m56 44 g663 112 m65 46 m57 45 p637 131 end
Action example:
m45: !move_base(Table4, Table5)
g552: (!grasp_object(Drink5, Table5, Trayarea2)
p237: (!place_object(Drink2, Trayarea3, Table7)

FIGURE 6. Result in delivery domain

in Figure 6. There were 12 actions to be executed for robot with PlanRM to accomplish the tasks, and spent totally 918 seconds, while 14 actions with Pyhop and totally 1053 seconds.

6. **Conclusions.** In this paper, we propose a planning approach for robot multitasking, which is based on hierarchical decomposing and employs an action parallelization process to optimize the plan to reduce the number of its actions and increase its efficiency on accomplishment of the tasks. As shown in the experiments, PlanRM, based on the proposed approach has made obvious progress compared with traditional approaches. The contributions are summarized as follows: extending model of HTN planning for RMP with the definition of a plan constituting with action cliques in which actions are compatible for executing in parallel; presenting the adapted algorithm of decomposing for the extended model and the re-planning algorithm based on partial backtracking; introducing an optimization method for action parallelizing, in which the optimization is modelled

as a maximum weighted clique problem; developing the planner of PlanRM, and proving its feasibility in experiments both in simulation and integrating it in a physical mobile robot. However, the approach and its planner have also some drawbacks, for example, our approach is not feasible for multi-agent system by now discussed, and we would say it is still at the beginning of its work in robot planning on task level. The attempt of integrating it in real robot and getting over the uncertainty of the real world is the bigger challenges.

Acknowledgment. This work is partially supported by National Natural Science Foundation of China (No. 61210013 and 61473278), Shenzhen Fundamental Research Funds (No. JCYJ20120615140035654), Guangdong Science and Technology Scheme (No. 2014A0 20215027), Research and Development Project of Guangdong Key Laboratory of Robotics and Intelligent Systems (No. ZDSYS20140509174140672). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] Y. Sakagami et al., The intelligent ASIMO: System overview and integration, *2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol.3, pp.2478-2483, 2002.
[2] M. Beetz et al., A roadmap for research in robot planning, *Technical Delivery of the European Network of Excellence in AI Planning*, 2003.
[3] D. McDermott, Robot planning, *AI Magazine*, vol.13, no.21, pp.55-79, 1992.
[4] J. Ni et al., An improved VFF approach for robot path planning in unknown and dynamic environments, *Mathematical Problems in Engineering*, 2014.
[5] S. M. Chen et al., Multirobot FastSLAM algorithm based on landmark consistency correction, *Mathematical Problems in Engineering*, 2014.
[6] Y. S. Lee and S. B. Cho, A hybrid system of hierarchical planning of behaviour selection networks for mobile robot control, *International Journal of Advanced Robotic Systems*, 2014.
[7] L. Einig et al., Parallel plan execution and re-planning on a mobile robot using state machines with HTN planning systems, *2013 IEEE International Conference on Robotics and Biomimetics*, pp.151-157, 2013.
[8] E. D. Sacerdoti, Planning in a hierarchy of abstraction spaces, *Artificial Intelligence*, vol.5, no.2, pp.115-135, 1974.
[9] L. P. Kaelbling and T. Lozano-Perez, Hierarchical task and motion planning in the now, *2011 IEEE International Conference on Robotics and Automation*, pp.1470-1477, 2011.
[10] P. R. Östergård, A new algorithm for the maximum-weight clique problem, *Nordic Journal of Computing*, vol.8, pp.424-436, 2001.
[11] F. Cazals and C. Karande, A note on the problem of reporting maximal cliques, *Theoretical Computer Science*, vol.407, pp.564-568, 2008.
[12] A. Hagberg et al., High-productivity software for complex networks, *NetworkX*, http://networkx.github.io/, 2014.
[13] R. Sedgewick, *Algorithms in C, Part 5: Graph Algorithms*, 3rd Edition, 2001.
[14] D. Long and M. Fox, The 3rd International Planning Competition: Results and analysis, *Journal of Artificial Intelligence Research*, vol.20, pp.1-59, 2003.