Proc. Workshop on Integrating Planning, Scheduling, and Execution in Dynamic and Uncertain Environments pp. 74-80, Pittsburgh, PA, June 1998.

Scheduling Parallel Execution of Planning and Action for a Hierarchically-Decomposable Planning Problem

Jun Miura and Yoshiaki Shirai
Dept. of Computer-Controlled Mechanical Systems, Osaka University
email: jun@mech.eng.osaka-u.ac.jp
URL: http://www-cv.mech.eng.osaka-u.ac.jp/~jun/

Abstract

This paper proposes a novel method to schedule parallel execution of planning and action. The method is for a class of planning problems which are hierarchically decomposed into two subproblems: (1) determining the next subgoal and (2) determining and executing an action sequence to achieve the subgoal. In this problem class, the upper-level planning process can be viewed as a process of gradually reducing the subgoal candidates towards the final commitment to one subgoal. Using criteria on deciding if a ground-level action sequence is consistent with the remaining subgoal candidates (*consistency criterion*), and on when to commit to one subgoal (*commitment criterion*), an appropriate action sequence is selected and executed while the upper-level planning process is still continuing. Preliminary experimental results including the comparison with a sequential method show that the proposed method is promising.

Introduction

Resource limitation and uncertainty are two important issues in planning for an agent in the real world. Since planning under uncertainty is usually costly, the limitation of computational resources tends to be critical. Controlling the planning process by explicitly considering the planning cost certainly improves the overall efficiency (Russell & Wefald 1991). Further improvement would be made possible by scheduling parallel execution of planning and action.

This paper is concerned with a vision and motion planning for a mobile robot in a *known* but *uncertain* environment. Fig. 1 shows an example planning problem treated in this paper. A vision-guided robot, which has a rough map of the environment, is going to the destination while avoiding obstacles. There is a route which passes the narrow space (we call it the *gate*); however the passability of the gate is initially unknown due to the uncertainty of vision. The detour passing through the hallway is known to be passable, although it is longer. The robot estimates the gate width with vision to determine the passability. Such a situation is quite usual; for example, in a typical office environment, the position of desks, chairs, and other furniture are roughly known, while their exact positions are uncertain; some chairs may block the robot from taking a certain path to the destination.

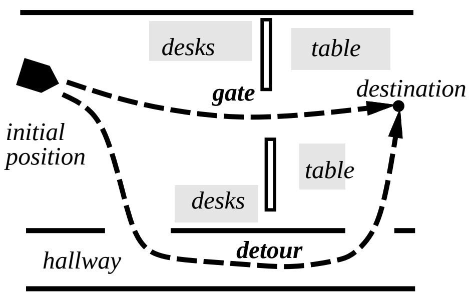


Fig. 1: An example problem.

The objective of planning here is to determine a sequence of observation points which leads the robot to the destination efficiently. We have been developing planning methods for this problem, in which several computational trade-offs are considered such as the one between the cost of visual recognition and the effect of visual information (Miura & Shirai 1993; 1997a) and the one between the planning cost and the plan quality (Miura & Shirai 1997b).

These methods are, however, *sequential*; that is, the planner runs while the robot is stopping, and once the next observation point is selected, the robot moves there. This sequential activation of planning and action is repeated until the robot reaches the destination. Since there is no physical limitation on executing planning and action in parallel, if we can properly schedule their parallel execution, more efficient operation of the robot will be realized.

Thus, this paper proposes a novel scheduling strategy which determines an appropriate action according to the information on the current planning process, such as what plan candidates are now under investigation, how long the current planning process will take, and what results will come out.

To solve the problem shown in Fig. 1, we hierarchically decompose the problem into the following two subproblems: (1) determining the next subgoal (i.e., the next observation point) and (2) determining and executing an action sequence to achieve the subgoal (i.e., the movement to the next observation point). The proposed strategy can basically be applied to a class of problems which are formulated in a similar manner.

Related Works

Realtime search algorithms (e.g., RTA* by Korf (1990)), which interleaves action selection with a depth-limited search and action execution, can be used for parallelizing planning and action; a simple scheduling strategy is to execute a selected action while searching for the next action. Nourbakhsh (1997) proposed criteria for controlling the search depth for interleaving. Goodwin (1994) explicitly considered if planning and action should be executed in parallel; his method compares two options, pure planning and planning while acting, in terms of the cost improvement rate, and if the latter seems better, the current best action is started. These methods deal with a class of problems in which a selected action is uniquely interpreted by the execution subsystem; no further planning for realizing an action is considered.

Zelek (1996) proposed a method of executing path planning and path execution in parallel for a sensor-based mobile robot navigation. Multiple path generators, which use maps with different spatial resolutions, run in parallel while the robot is moving. As long as the path is generated within one action control cycle, the path generated in the finer resolution is used. This method is also for the above-mentioned problem class.

If a planning problem is hierarchically decomposed into subproblems, as in the case of our vision and motion planning problem, planning in multiple levels and action in the ground level can be executed in parallel. Nourbakhsh (1997) proposed an abstraction-based planning method. The original problem is mapped into abstract problem spaces, and once a solution is obtained in one space, it is given to the lower-level planner as the subgoal; after the subgoal is achieved, the lower-level process waits for the next subgoal information from the higher-level.

Basic Scheduling Strategy

The basic idea in the proposed method is as follows. The higher-level planning process of determining subgoals for the ground-level can be viewed as a process of gradually reducing the subgoal candidates towards the final commitment to one subgoal (Kambhampati, Knoblock, & Yang 1995). If there are some ground-level actions which are consistent with (or, do not largely conflict with) remaining subgoal candidates, such actions can be performed while the higher-level planning process is still continuing.

In our vision and motion planning problem, we define a *subgoal* as an observation point and *action* as a movement towards the next observation point. An action can be executed in parallel with the planning of the next observation point as long as it is consistent with the remaining subgoal candidates.

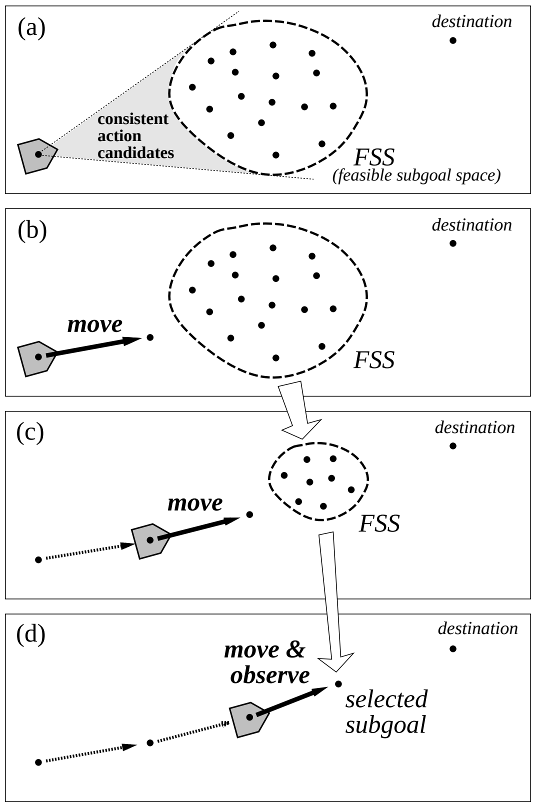


Fig. 2: Gradual reduction of subgoal candidates (FSS) and selection of actions.

The above idea can be schematically explained in Fig. 2. Fig. 2(a) shows an initial state. The initial set of feasible subgoal candidates is derived, which are collectively called an *FSS (feasible subgoal space)*. Once an FSS is calculated, a set of actions which are consistent with the FSS is selected (see the shaded area in Fig. 2(a)). Then, among the candidate actions, the best one is selected and executed (see Fig. 2(b)). As the planning process proceeds, the FSS is reduced and actions are selected and executed repeatedly (see Fig. 2(c)). Finally the planning process selects one subgoal, and, a few moments later, the robot reaches the selected observation point and observes the environment (see Fig. 2(d)). If we employ the sequential method, the robot is still at the initial position when the commitment to the subgoal is made. Thus, the distance which the robot travels until the commitment is made is the merit gained by the parallel method.

Since the planning and the action processes, in principle, can be executed asynchronously and in parallel, the space of possible schedules could be too huge to

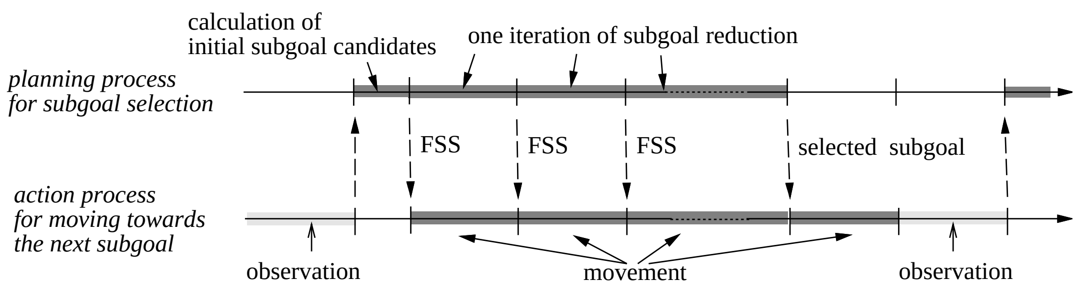

calculation of initial subgoal candidates
one iteration of subgoal reduction
planning process for subgoal selection
FSS
FSS
FSS
selected subgoal
action process for moving towards the next subgoal
observation
movement
observation

Fig. 3: Time chart of planning and acting processes.

search. Thus, we limit the timing of changing actions only to the end of each candidate-reducing cycle of the planning process. The time chart of the planning and the action processes would look like Fig. 3.

To implement the above scheduling strategy, the following should be provided:

- A planning process which gradually reduces the candidates.
- A criterion to commit to one subgoal (or one plan). We call it *commitment criterion*.
- A criterion to evaluate the consistency between an action candidate and an FSS. We call it *consistency criterion*.

The planning process will be described in the next section. The concrete algorithm including the criteria for an example problem setting will then be described.

Planning Process

We use a modified version of the previously developed planning method (Miura & Shirai 1997b) as the planning part (subgoal determination part). This section briefly explains the modified planning method.

Plan Representation

A state is represented by the current estimate of the gate width and the current robot position. Due to the uncertainty in observation results, the robot cannot determine the gate width deterministically but obtains its probabilistic distribution (Miura & Shirai 1997a). After an observation, the robot classifies the state of the gate into one of the three categories (*passable*, *impassable*, and *unknown*) according to the relationship between the probabilistic distribution and the robot width (see Fig. 4).

Since the actual state after an observation depends on the observation result and cannot be determined beforehand, a subplan is generated for each possible state. Fig. 5 shows an example plan for the problem shown in Fig. 1. Such a plan is represented by a special AND/OR tree which has one OR node at each level; an OR node corresponds to a selected action; an AND node corresponds to a possible state. The quality of a plan is measured in terms of its execution cost, which is the *expectation* of the total execution time for movement *and* observation.

Iterative Refinement Formulation

To trade the planning cost against the plan quality, we formulate the planning process as an anytime iterative refinement process (Boddy & Dean 1989); i.e., the

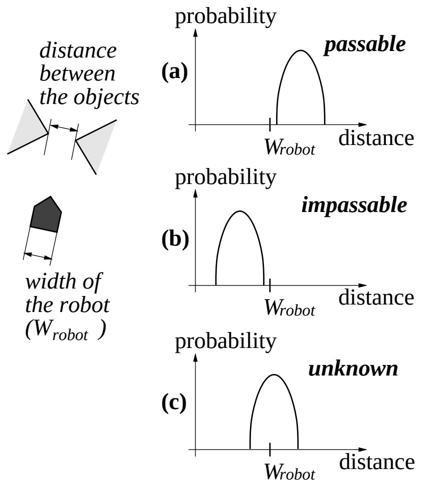


Fig. 4: Three possible state of the gate.

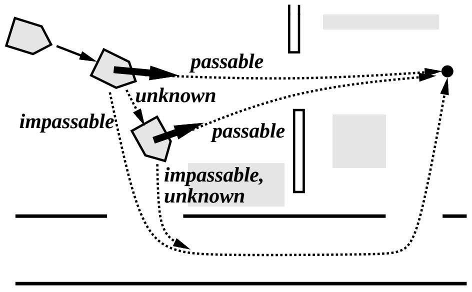


Fig. 5: An example plan for the problem shown in Fig. 1. Dotted arrows indicate possible movements after observation. Bold arrows indicate observation of the gate.

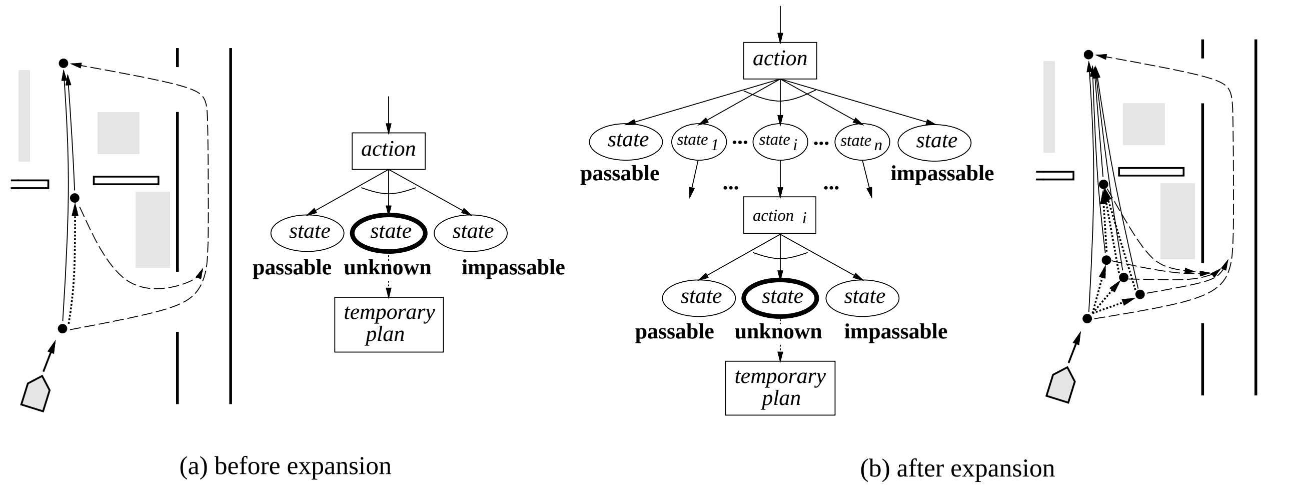


Fig. 6: Expansion of an open node of a plan candidate and its corresponding situation. Ellipses drawn with bold lines indicate open nodes. In the leftmost and the rightmost figures, solid arrows, dashed arrows, and dotted arrows correspond to movement when the gate's state is passable, impassable, and unknown, respectively.

planner searches *the space of feasible plans* (executable plans) for the final plan. This formulation entails an *easily-obtainable* feasible plan for any open node. We here use the following one:

The robot moves from the current position to the position just before the gate[1]*. If the gate is passable, the robot passes it; if not, the robot takes the detour from that position.*

Each plan candidate has the *temporary cost*, C^{temp}, which is obtained by *temporarily* assigning this feasible plan to all of its open nodes.

In the refinement process, all of open nodes of each plan are expanded. Before expansion, an unknown state is treated as an open node and has a feasible plan with it (see Fig. 6(a)). The expansion of the open node consists of discretizing the range of possible gate width for the node with some *granularity*, searching for the best action for each discretized state, and assigning the feasible subplan to newly generated open nodes (see Fig. 6(b)).

Suppose we can predict the plan improvement (i.e., cost reduction) ΔC of a plan candidate, which will be obtained by expanding all of its open nodes[2]. Then, the *new cost* C^{new} after expansion is given by subtracting the plan improvement from the temporary cost:

$$C^{new} = C^{temp} - \Delta C. \quad (1)$$

Let C_{FP^*} be the cost of the *incumbent* FP^* (the best feasible plan among those which have been obtained so far). During planning, the plan candidates are kept whose new costs are less than C_{FP^*}. The plan candidate p^* which has the minimum new cost is determined by:

$$p^* = \arg\min_p C_p^{new}. \quad (2)$$

C_{FP^*} and $C_{p^*}^{new}$ will be used later for defining the commitment criterion (i.e., for determining when to stop the iterative refinement process).

Note that although the planning process iteratively refines plan candidates, the refinement at deeper levels than the first level is for comparing competing candidates for the first subgoal.

Scheduling Algorithm for 1-D Planning Problem

Problem Description

Fig. 7 shows an example planning problem. This is a simplified 1-D version of the problem shown in Fig. 1. The robot is initially at $\boldsymbol{x}_0$; $\boldsymbol{x}^*$ is the zero-uncertainty point used in feasible plans. The next observation point is selected on the line segment connecting $\boldsymbol{x}_0$ and $\boldsymbol{x}^*$. If the robot decides to take the detour from an observation point, it turns back and passes $\boldsymbol{x}_0$ towards the detour.

FSS and Consistent Action Candidates

For each plan candidate p, we calculate temporary cost C_p^{temp} and predicted plan improvement ΔC_p, thus obtaining new cost C_p^{new} using Eq. (1). The plan candidates whose new costs are less than the cost C_{FP^*} of the incumbent compose a set of feasible plan candidates. The current FSS is constructed as a set of the first subgoals (observation points) of feasible plan candidates.

[1]At this position, the robot is assumed to be able to measure the gate width without uncertainty; this position is called the *zero-uncertainty point*, indicated as x^*.

[2]A prediction method will be described later.

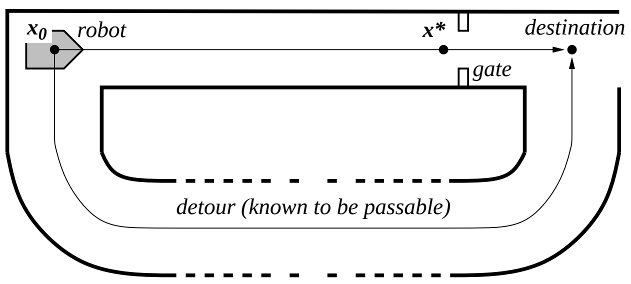


Fig. 7: 1-D Planning problem for simulation.

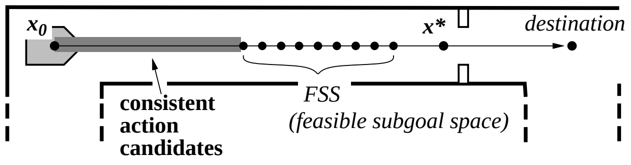


Fig. 8: FSS and candidate actions.

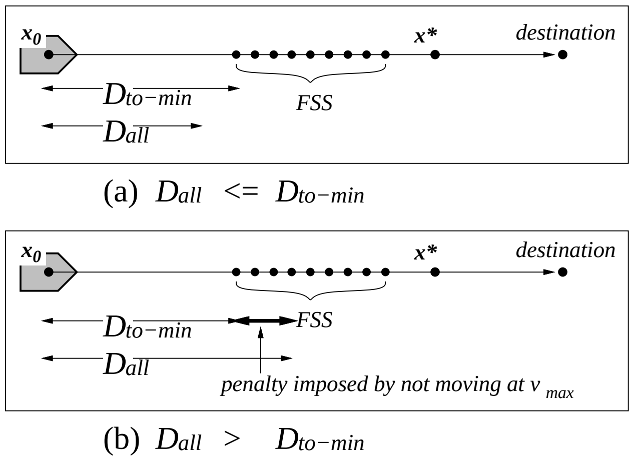
In this 1-D problem, an FSS collapses into a 1-D segment as shown in Fig. 8. Thus, the *consistency criterion* here is that an action is consistent with an FSS if it is to move to a point before the FSS. Using this criterion, consistent candidate actions are generated as shown in the figure.

Determining Best Action

The degree of parallel execution is maximized if the ground level action finishes just at the end of the current refinement step of the planning process. Thus, to determine the best action, explicit estimation of planning cost is necessary. We estimate the time required to perform the next refinement step as follows. For each candidate p, we calculate the cost (i.e., time) of expansion C_p^{exp}, which is obtained as the sum of expansion costs for p's open nodes. From all such costs, the time for the next refinement step is given by

$$C_{all}^{exp} = \sum_p C_p^{exp}. \tag{3}$$

Then we calculate the two distances: D_{all} and D_{to-min}. D_{all} is the distance covered if the robot moves at its maximum speed v_{max} for the duration of C_{all}^{exp}. D_{to-min} is the distance to the nearest feasible candidate, that is, the distance to the leftmost point of the FSS in Fig. 8. The relationship between D_{all} and D_{to-min} has an important role in the scheduling algorithm. Possible cases are enumerated as follows.
(Case 1): If D_{all} is less than or equal to D_{to-min} (see Fig. 9(a)), since the robot finishes the next refinement step before arriving at the FSS, the robot can move at v_{max} while performing the refinement. After finishing the refinement, a new action is selected based on the updated FSS.



x_0
x^*
destination
D_{to-min}
FSS
D_{all}
(a) D_{all} <= D_{to-min}
x_0
x^*
destination
D_{to-min}
FSS
D_{all}
penalty imposed by not moving at v max
(b) D_{all} > D_{to-min}

Fig. 9: Two possible relationship between D_{all} and D_{to-min}.

(Case 2): If D_{all} is larger than D_{to-min} (see Fig. 9(b)), performing the next refinement step instead of executing the current incumbent (FP^*) has a loss; namely, to perform the next refinement step, the robot has to move at a slower speed than v_{max} so that the robot does not pass any of candidate observation points; on the other hand, the robot can move at v_{max} in executing FP^*[3]. Thus, we compare C_{FP^*} and the minimum of the new cost $C_{p^*}^{new}$ with a certain penalty being imposed on the latter. The penalty is calculated as the time needed to move the distance $D_{all} - D_{to-min}$ at v_{max}. Using this penalty, the decision is made as follows.
(Case 2-a): If $C_{p^*}^{new} + penalty$ is less than or equal to C_{FP^*}, the robot moves to the nearest candidate while performing the refinement. After finishing the refinement, a new action is selected based on the updated FSS.
(Case 2-b): If $C_{p^*}^{new} + penalty$ is larger than C_{FP^*}, the robot stops the iterative refinement process, and executes FP^*. This is the *commitment criterion*. Execution of FP^* is composed of moving to the specified observation point and observing the gate. If the passability is decided after the observation there, the robot passes the gate or takes the detour towards the goal. If the passability is still unknown, a new action is selected based on the updated FSS.

Simulation Results

Prediction of Plan Improvement

The improvement ΔC_p which is to be obtained by expanding open nodes of a plan candidate is predicted

[3]This is true under the assumption that D_{all} is smaller than the distance to the observation point specified by FP^*.

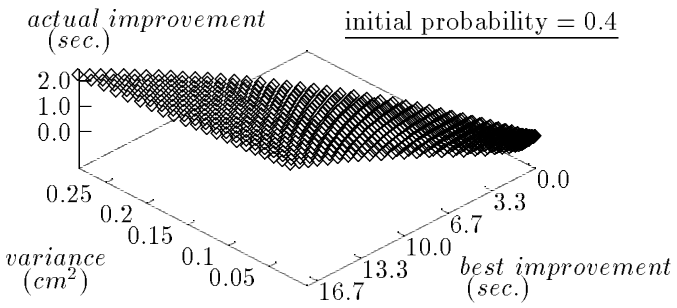


Fig. 10: Plot of actual improvements in the 2-D parameter space.

using the learned data collected by solving actual planning problems with various parameter settings. In the previous work (Miura & Shirai 1997b), we first constructed a generalized expression with several coefficients to represent the relationship between the problem parameters and the plan improvement, and then adjusted the coefficients using the experimental data.

In this paper, however, we use a simpler method. That is, in the learning phase, we store the plan improvements in the space of several problem parameters. In the planning phase, for a specific combination of these parameters of an actual problem, the plan improvement is predicted by using a nearest neighbors method using k neighbors, namely, by selecting k nearest neighbors from the learned data, and then by calculating the mean of the corresponding stored plan improvements. If a plan candidate has multiple open nodes, we use the weighted sum of such retrieved plan improvements, using the probability of reaching each open node as the weight.

From the calculation results of actual plan improvements, we found that the plan improvement is well described by two parameters, an upper bound of the improvement (Miura & Shirai 1993) and the variance of the probabilistic distribution of the gate width; we also use the probability of the gate being passable as another parameter to switch the learned data set used. Fig. 10 shows the plot of actual improvements in the 2-D parameter space, for the case that the probability of the gate being passable is 0.4. In the planning phase, an appropriate data set is selected according to the actual probability.

Results

On the 1-D range between initial position $\boldsymbol{x}_0$ and zero-uncertainty point $\boldsymbol{x}^*$, we set grid as candidates of the next observation point. Fig. 11 shows some simulation results, in which (i) the movement of the robot in a 2-D time-distance space and (ii) the gradual reduction of the subgoal candidates (i.e., FSS) until the robot reaches the next observation point are indicated. In the figures, a bold arrow indicates the movement of the robot; a filled circle on the movement indicates the end of one refinement step; the shaded horizontal strip corresponding to the filled circle is an FSS and its nearest point may be the destination of the next action. The FSS is gradually reduced to the final commitment to the next observation point (FP^*).

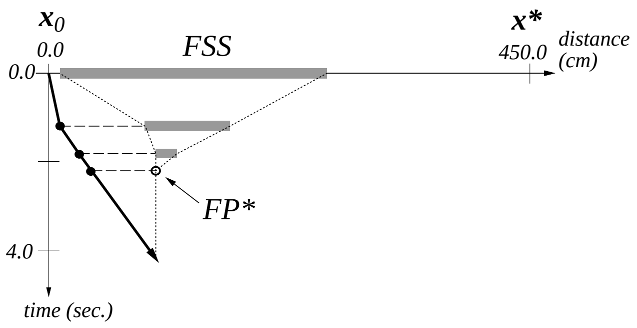
In the three figures, the initial probability of the gate being passable is the only difference; the probabilities are 0.1, 0.25, and 0.4 for cases (a), (b), and (c), respectively. As the initial probability increases, the position of the FSS becomes nearer to $\boldsymbol{x}^*$; thus the speed of the first action becomes larger. If the probability is much larger (e.g., 0.8), the FSS is initially an empty set, and the initial incumbent is immediately executed.

Fig. 12 shows another simulation result, in which the time saved by using the parallel scheduling method instead of the sequential one is plotted for various initial probabilities. The effect of the parallel scheduling is mainly determined by the number of plan candidates examined. If the initial probability is close to 0.0 or 1.0, the situation is almost certain and the number of plan candidates is small, therefore, little planning effort is needed to find the best solution. If the probability is in the middle of the range, however, the situation is more uncertain and much planning could be needed; for such a case, the parallel scheduling of planning and action is more effective. Fig. 12 roughly shows such a tendency.

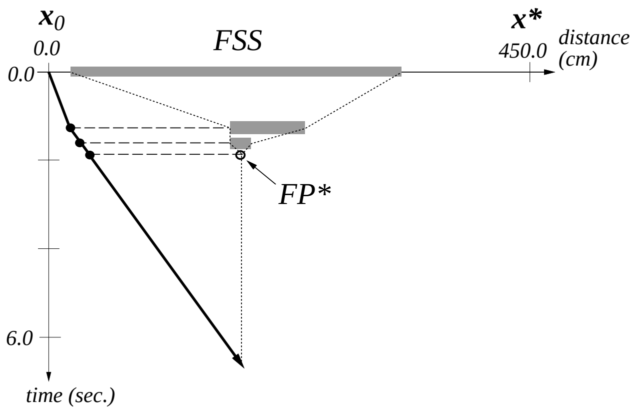
Concluding Remarks

We have proposed a novel method to schedule parallel execution of planning and action for hierarchically-decomposable planning problems. The planning part is realized by an iterative refinement planner. By considering the current set of plan candidates and the expected time for the next refinement step, the action is selected which is consistent with plan candidates and the most efficient. The consistency criterion and the commitment criterion are the important concepts in the method. Simulation results show that the proposed method is promising.

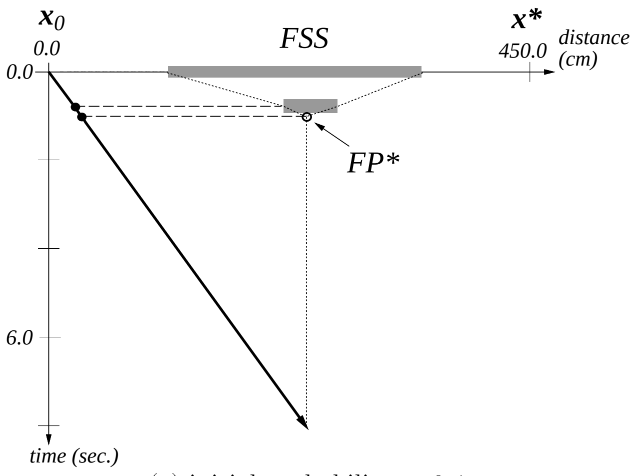
We are now applying the method to 2-D planning problems like the one shown in Fig. 1. Although the structure of the algorithm designed for the 1-D problem could be used unchanged, the consistency criterion needs to be modified, because the action towards one observation point is not necessarily on the direction towards another observation point. The set of consistent actions would look like the one shown in Fig. 2(a). If the FSS forms two (or more) clusters at different positions, however, there may be no consistent actions, and the robot may have to continue planning while stopping. The design of *consistency criterion* considering various cases would be the key to overall efficiency of the method.



(a) initial probability = 0.1.



(b) initial probability = 0.25.



(c) initial probability = 0.4.

Fig. 11: movement of robot and the reduction of FSS.

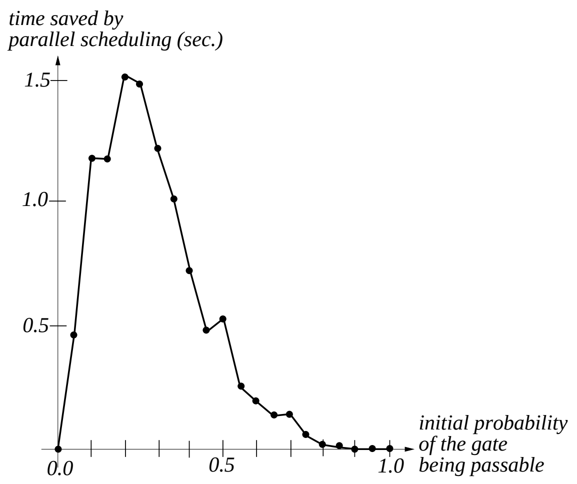


Fig. 12: Saved time by parallel scheduling. The time for moving from x_0 to x^* at v_{max} is 15 (sec.).

Acknowledgments

This work is supported in part by Grant-in-Aid for Scientific Research from Ministry of Education, Science, Sports, and Culture, Japanese Government and by Tateishi Science and Technology Foundation, Kyoto, Japan.

References

Boddy, M., and Dean, T. 1989. Solving time-dependent planning problems. In *Proceedings of IJCAI-89*, 979–984.

Goodwin, R. 1994. Reasoning about when to start acting. In *Proceedings of the 2nd Int. Conf. on Artificial Intelligentce Planning Systems.*

Kambhampati, S.; Knoblock, C. A.; and Yang, Q. 1995. Planning as refinement search: A unified framework for evaluating design tradeoffs in partial-order planning. *Artificial Intelligence* 76:167–238.

Korf, R. 1990. Real-time heuristic search. *Artificial Intelligence* 42:189–211.

Miura, J., and Shirai, Y. 1993. An uncertainty model of stereo vision and its application to vision-motion planning of robot. In *Proceedings of the 13th Int. Joint Conf. on Artificial Intelligence*, 1618–1623.

Miura, J., and Shirai, Y. 1997a. Vision and motion planning for a mobile robot under uncertainty. *Int. J. of Robotics Research* 16(6):806–825.

Miura, J., and Shirai, Y. 1997b. Vision-motion planning for a mobile robot considering vision uncertainty and planning cost. In *Proceedings of the 15th Int. Joint Conf. on Artificial Intelligence*, 1194–1200.

Nourbakhsh, I. 1997. *Interleaving Planning and Execution for Autonomous Robots.* Kluwer Academic Publishers.

Russell, S., and Wefald, E. 1991. *Do The Right Thing.* The MIT Press.

Zelek, J. S. 1996. Performing concurrent robot path execution and computation in real-time. In *1996 AAAI Fall Symposium on Flexible Computation in Intelligent Systems: Results, Issues and Opportunities.*