Probabilistic, Distributed, Multi-agent Planning in Uncertain Environments

Jamieson Schulte
Center for Automated Learning and Discovery
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
USA

February 5, 2001

Abstract

We propose research into a new algorithm for distributed planning among multiple agents for movement in an uncertain graph-like environment. The planning method is a distributed probabilistic search in shared belief space by coordinated agents. The algorithm outputs a partial contingency plan for each agent, which is meant to approximate the incomputable optimal solution. Due to low inter-agent communications overhead, the planning algorithm is readily parallelizable and can scale to many agents and goal states. By focusing on states that (1) are more likely to occur and (2) bring the agent closer to states from which it can contribute to a global reward, the planning algorithm should be able to achieve good performance using relatively little computation time. The proposed algorithm has an any-time property, which means that it can output a set of contingency plans at any point during its search. This would allow mobile agents to begin execution of plans while continuing to devote computation resources to further refinement of later actions.Early results based on a similar algorithm show that the method should scale to many hundreds of agents and goals, and can outperform a previous algorithm that centrally assigns each agent to a single goal. Because the properties of this type of search are not yet well-known, further research is needed to determine when and how such a planner should be used. One important area of investigation is comparison with more general probabilistic planners such as those that work in classical STRIPS domains. To make performance comparisons, the project would work collaboratively with other researchers to produce a uniform environment for testing planning algorithms in the target domain. If successful, the algorithm produced by the research could be applied to a variety of real-world problems including emergency relief following a natural disaster, multi-robot movement and exploration, and package delivery in dynamic environments.

1 Introduction

1.1 Problem

The proposed research addresses the problem of finding paths for multiple mobile agents to multiple goals in an uncertain environment that maximizes a reward acquired upon reaching each goal. This is a multi-agent instance of what has been called the Canadian Traveler's Problem, in which an agent must travel from one point to another on a graph in which each edge has some known probability of being traversable. In the multi-agent case the environment is the same, but there are many agents and goal states.

The result of planning is a contingency plan for each agent that maps agent positions and knowledge about the world to movement actions. It is known that the optimal solution to the single-agent problem is NP-hard [BK99], and it follows that the problem solution becomes significantly more complex when there exist multiple agents and goal states. Thus, the goal of the proposed research is to find an algorithm that approximates an optimal solution using significantly less computation.

The planning problem is made difficult by several factors:

- Developing a complete contingency plan requires computation exponential in the size of the graph.
- The number of contingency plans is exponential in the size of the graph.
- Assignment of multiple agents to multiple goals is exponential in the number of agents.
- If planning is distributed, there must be a mechanism that ties local decisions to their effect on global reward.

In summary, any feasible solution to the problem must use a small amount of computation to select from a large number of plans, each of tremendous complexity. This must be done in a way that achieves a good global reward while using limited communications bandwidth between the distributed planning units.

1.2 Deliverables

The research, if successful, will produce an algorithm for distributed planning under the circumstances outlined above. The primary advantages of the proposed algorithm are:

- Scalability: the computational, memory, and inter-agent communication requirements will permit implementations involving thousands of agents in real-world environments. Low communications overhead means that planning computation can be parallelized and distributed among the actual agents.
- Any-time planning: the planning algorithm can be allotted an open-ended range of computation cycles and return a plan at any point during its operation. More computation will allow improved plans, but require more time.
- Improvement over naive approaches: the performance of the resulting algorithm will be demonstrably better than that achieved using a similar amount of computation and making obvious extensions to single-agent and deterministic planners.

To test the performance and behavior of the algorithm, and to allow accurate comparison to other methods, a software simulation environment will be developed which can position agents in a virtual environment and record their performance in different environments.

1.3 Practical Impact

The proposed research will address problems at the core of many real planning problems, including:

- Disaster relief: following a natural disaster, it is imperative to bring supplies to the affected people as quickly as possible. However, the post-disaster state (traversability) of roads and bridges in the damaged regions is only partially known. Previous work by the author has been oriented toward a disaster relief scenario in Honduras following hurricane Mitch in 1998.
- Robot motion planning and mapping: robot teams in real, populated environments contend with blocked corridors, closed doors, crowds, and other stochastic impediments. In robot mapping, an unknown environment must be explored efficiently [TFB00].
- Package delivery: many businesses offer high-speed package delivery in urban areas using multiple deliverers to many destinations, and must deal with uncertain road conditions (due to traffic, road work, etc.)

Each of the listed planning applications could be helped by an algorithm that efficiently computes a good approximation to the optimal multi-agent plan.

2 Prior Work

Agent planning in graph-like environments has been extensively studied since the birth of AI research. Typically however, previous investigations have focused on single agents or fully observable environments. By single agent, it is meant that planning occurs from a single position on the graph. A fully observable environment is one in which the graph structure and traversability is completely known prior to planning.

The main focus of planning research in recent decades has been on the abstract STRIPS-style domain that is a superset of the mobile agent problem. Examples are Graphplan[BF97], IPP[KNHD97], and STAN[LF98]. Graphplan has been extended to stochastic domains, namely that representable by an MDP[BL99]. Other general probabilistic planners exist, such as HSP[BG00] and CMBP[CR00]. HSP automatically finds search heuristics that are useful in the problem domain. While such planners can be applied to our multi-agent domain, we expect to obtain an advantage by

1. distributing the computation among many agents, and
2. making use of information provided by a spatial interpretation of the graph.

Table 1 samples some of the work that has been done in these different areas with some application to mobile agents. The A* algorithm [Nil82] is widely used, but does not address multi-agent and partially observable situations. HSP and Probabilistic Graphplan are designed for stochastic environments, but do not generalize to multi-agent problems. Work on multi-agent dispatching[LST01] addresses the same problem domain and takes a similar approach, but does not address issues of scalability that arise with large numbers of agents.

		Environment	
		Fully Observable	Partially Observable
Agents	Single	A* heuristic search (Nilson)	HSP, Probabilistic Graphplan (Bonet & Geffner), (Blum & Langford)
	Multiple	concurrent interacting plans (Boutilier & Brafman)	multi-agent dispatching (Lita, Schulte & Thrun)

Table 1: A sample of prior work in agent planning in graph-like physical environments, organized according to number of agents and type of environment.

Some areas of research, such as interned packet routing share a great deal with the problem of interest, but in application do not have the luxury of computational time required to perform a

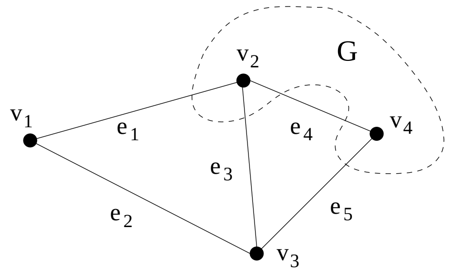

Figure 1: An small example graph environment.

planning step. Instead, these methods tend to have simple reactive behaviors sometimes coupled with intermittent sensing of the network environment [CBGF97] and machine learning[LB93].

While the planning literature offers a wealth of approaches to problems with a very different set of constraints, the solutions offered thus far do not take advantage of unique characteristics of the problem addressed here.

3 Approach

3.1 Background

The problem consists of finding contingency plans for many agents in an environment that can be modeled using a graph structure. To develop a clear picture of our approach, the basic concepts are outlined below. First, the model of the environment is described, followed by the agent and belief state representation, and a brief overview of heuristic search algorithms. Next the single-agent planning approach is outlined, followed by the proposed multiple-agent extension to this idea.

3.1.1 Environment Model

Agents are assumed to exist in a graph-like world that is represented as $W = \{V, E, G, P_s\}$, where V is the set of graph vertices and E is the set of edges that connect two vertices. $G \in V$ is the set of "goal" vertices. Each edge e_i has a length (or cost) l_i, and a binary state $s_i \in \{0, 1\}$. The state is a random variable that indicates whether the edge is traversable by an agent. We can collect the edge states into a vector $\mathbf{s} = [s_1 s_2 ... s_n]$ which we call the *world state*. The final property of the environment is a probability distribution P_s over all possible edge state configurations, $Pr(\mathbf{s} = \mathbf{y})$. When a graph is instantiated, a world state configuration is sampled from P_s and is assumed to remain fixed in this work. Figure 3.1.1 shows a very small example graph.

3.1.2 Agent Model

An agent exists at a position in the graph environment and, depending on its position and the graph state, is able to perform one of several actions. We model an agent's interaction with the world as a form of partially observable Markov Decision Process (POMDP), which have been described extensively in the learning and planning literature[KLC95, Lit95, PR95]. An agent's *belief state* consists of a position in the environment and belief about the states of edges. An agent may be positioned at a vertex or at any intermediate point on an edge. The agent belief is represented as a vector of *belief values* $B = [b_1 b_2 ... b_n]$ for the edges in the graph. A belief value, $b_i = E[s_i|obs]$, is the expected value of an edge state, given all observations made by the agent thus far.

The set of actions available to an agent depends on its position in the graph, and consist of movement along adjacent edges. If an agent attempts to move along an edge that is of unknown state, then it must travel half of the edge length to discover whether or not it is traversable. If it is, then the agent may move to either end of the current edge, and its belief is updated to reflect the knowledge that the edge is traversable. If not, then the only available action is to return to the starting vertex and update the belief value for the edge accordingly. Because an edge must be visited to learn its state, there is a sizable cost to observing the true state of the environment. Once information is obtained by one agent, it is broadcast to all others. Replanning occurs if existing agent plans cannot make use of the new information.

The purpose of planning is to reach goal states in as little time as possible. The first agent to reach a particular goal state g receives a time-discounted reward of the form

$$R_g(t) = e^{-\alpha t_g}$$

where t_g is the time taken to reach g, and α is a decay parameter that is adjusted to suit the characteristics of the problem. The global reward which we attempt to maximize in the multi-agent case, is the sum of rewards collected at each goal.

$$R = \sum_{g \in G} e^{-\alpha t_g}$$

Because the true state of the world is unknown, t_g is a random variable, and we can think only in terms of *expected* reward. In this context, the goal of planning is to maximize

$$E[R] = \sum_{g \in G} e^{-\alpha E[t_g]}$$

3.1.3 Heuristic Search

The most common form of planning by single agents in fully observable graph environments are heuristic search algorithms such as A*. The core idea of heuristic search is that of prioritized state expansions based on a heuristic function $h(s)$. The heuristic function provides an estimate of the remaining distance to the goal state from the current search state. If the heuristic is guaranteed to be less than or equal to the actual distance to the goal, then it is termed *admissible*, and it has been shown that this leads to a complete search.[Nil82] The closer that the heuristic comes to estimating the true distance to the goal state, the fewer the number of expansions that are required to complete the search.

When the environment is partially observable, we in most cases cannot hope to perform a complete search. Furthermore, it is not straightforward to account for the behavior of multiple agents moving to multiple goals in a single heuristic function. Thus it is necessary to make significant changes to the baseline heuristic search algorithms in order to address the problem of interest here.

3.2 Single-agent Planning

We now describe the approach that we expect to take as it would apply to single-agent planning before undertaking an elucidation of the more complex multi-agent case. Here it is assumed that there is a single agent and a single goal state, and the environment and agent are as just described. It is further assumed that edge states are statistically independent, so knowledge about the state of one edge provides no information about that of any other edge. The agent begins with a belief state that contains only the (true) probabilities that each edge is traversable, according to the environment's actual world state distribution.

Planning proceeds by building a search tree through prioritized expansion of search states that occur at the leaf nodes. The tree encodes multiple contingency plans in a single compact tree structure as needed to select the best plan discovered upon termination of the search. Each

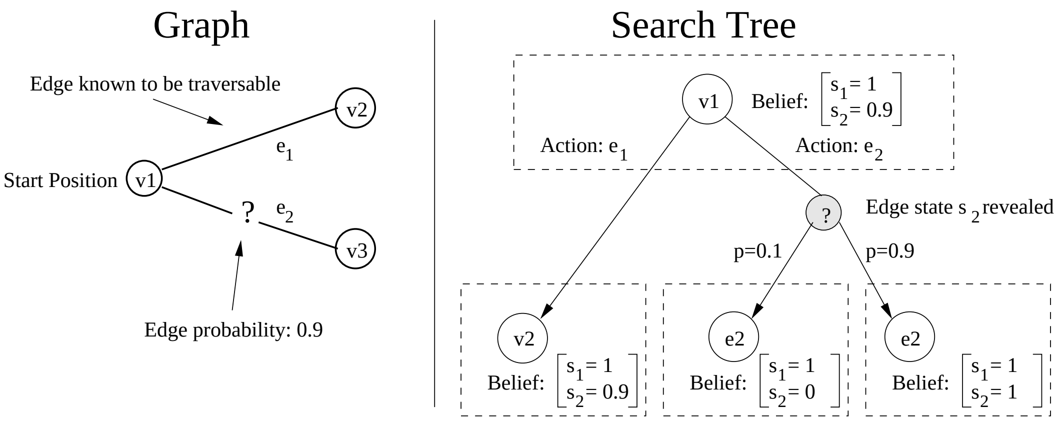

Figure 2: A pictorial representation of the search process. On the left is a very simple graph section with one edge of known state and another of unknown state. To the right is a diagram of the search tree for two possible actions available from vertex v_1. For the action that follows e_1, the agent arrives at v_2 with certainty and discovers no new information. The action that follows e_2, however, terminates with the agent on the edge having just learned whether it is traversable. In either case, the agent belief for s_2 is updated to reflect the new information.

node in the tree contains a belief state, and links to child nodes that represent new belief states corresponding to each possible action from that state, and the outcome given different world states. Figure 2 shows the overall structure of such a tree. Following a search, the tree can be analyzed to determine the estimated expected reward for each contingency plan and to return the plan that maximizes this value.

A probabilistic priority function $f(s)$ is used to order the expansion of search states. The function is an attempt to provide an optimistic estimate of the expected reward to be received by an agent passing through state s. The form parallels that used by the A* algorithm, but in a probabilistic context that attempts to estimate expected reward instead of total travel distance:

$$f(s) = p_s e^{-\alpha(g(s)+h(s))}$$

where p_s is the probability of reaching state s following the plan so far, $g(s)$ is the time required to reach s, and $h(s)$ is the estimate of expected time to reach the goal state.

Using this priority function, planning occurs as follows:

- Initialize the agent belief vector to reflect the current knowledge of the world state.
- Initialize an empty search priority queue.
- Initialize an empty plan tree.
- Put the agent starting belief state into the search priority queue.
- Repeat until the search queue is empty, or the algorithm has exceeded a set number of iterations:
  - Remove the top belief state from the search queue, and assign it to b
  - Update the plan tree to contain state b
  - For each action possible from b, add a new search state to the search queue, with priority determined by b, the action, and the heuristic function.
- Evaluate the plan tree and return the contingency plan that maximizes the expected future discounted reward.

3.3 Multi-agent planning

When there are multiple agents, it becomes necessary to consider global reward rather than a single agent's reward. Furthermore, additional agents and goals complicate estimates of reward because it may be unknown which agent arrives at a goal first. It is also important for planning to take into account the fact that some agents are better suited to reach a particular goal than others. Thus extensions must be made to the single-agent planning algorithm in an attempt to address these complications.

The primary direction of our proposed research is to communicate between agents their expected arrival times and probability of arrival at each goal. The search priority function can then be modified so that these factors are taken into account *during* planning, in order to influence agents to plan actions that will maximize the global reward defined earlier. Suppose that at any point in planning, each agent keeps an estimate of its expected time to reach each goal and broadcasts those values. Using this information, an agent can estimate its own expected contribution to the global reward for each available goal. Such an estimate can be used to prioritize the goals according to the agent's expected reward upon reaching them. From a prioritization of the goals, an agent can prioritize search states by weighting the expected distance of a state from each of those goals. Thus, agents can share a relatively small amount of information and make plans that (locally) optimize the global reward without central oversight. Thus far, our research has not uncovered the best way to do this. However, some preliminary experiments have been performed which use priority functions that share information between agents.

3.4 Preliminary Results

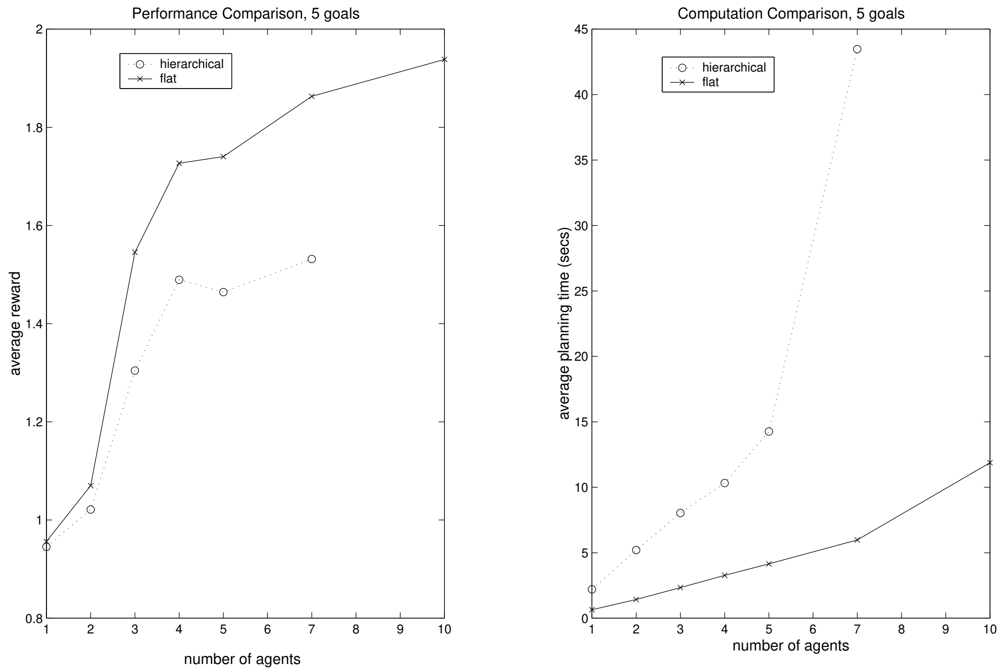

Figure 3: Comparison of an initial implementation of the proposed algorithm (called "flat") to a hierarchical method that uses a central dispatcher to assign agents to goals.

An *ad hoc* multi-agent planning algorithm that is based on the ideas presented above has been implemented and subjected to basic testing in a simulated environment. The current implementation does not cover all of the desirable qualities of a probabilistic multi-agent planner, but it provides a comparison to a previous technique that does not share information between agents. In an initial experiment, the algorithm generated plans for 1,500 agents in several minutes on a single machine. This provides good evidence that it can scale well, especially in implementations involving multiple processing units.

In a second experiment, the test algorithm is compared to a "hierarchical" planner, which has each agent make a plan to each goal, and performs an exhaustive search to find an agent-goal assignment that maximizes global reward under the assumption that the agent plans operate independently. Results for both global reward and computation requirements in simulation are shown in Figure 3.

Two major weaknesses affect the performance of the hierarchical planner, both of which are addressed in the proposed research. First, each agent makes a plan to each goal without any information about the other agent plans. Second, the central dispatcher assigns agents to goals without considering that the success or failure of each of the agent plans may be heavily dependent upon each other. The hierarchical approach also faces computational problems since each agent plans to every goal, and the dispatcher must evaluate on the order of $|G|^{\#ofagents}$ agent-goal combinations to find the best pairings.

The results show that not only can planning time be reduced to a quantity nearly linear in the number of agents, but that performance can be improved.

4 Work Plan

The proposed research would be carried out over the next ten weeks:

- **March 9** Develop a probabilistically sound estimate of expected reward for agents. From this, an estimate of agent contribution to global reward can be derived, which will be the basis of a new search heuristic and priority function.
- **March 16** In order to rigorously test the performance of the new algorithm under a variety of conditions against other planning methods, a standard simulation environment will be created. The simulation environment will allow careful analysis of the algorithm's behavior in different circumstances, and will allow comparison to other planning methods.
- **April 13** The algorithm will be refined using any new information discovered during the previous testing step.
- **April 27** It will be important to show that any new algorithm designed specifically for the stochastic mobile agent domain outperforms more general existing planners such as HSP and Probabilistic Graphplan. Experiments will be performed to evaluate these planners though considerable effort may be involved in representing the multi-agent planning problem in a way that can be used by them.
- **May 7** Finally, a research paper and in-class lecture will be written to describe the results of the work in order to disseminate this knowledge to the planning community.

5 Summary

We propose research into a new algorithm for distributed planning among multiple agents for movement in an uncertain graph-like environment. Preliminary results show that the approach can scale to hundreds of agents and generate plans that outperform those produced by an earlier algorithm. The algorithm to be researched, a distributed probabilistic heuristic search, has many properties that are improvements over existing methods.

The basic problem addressed by this research is common to many scenarios in which physical agents must reach a set of destinations as quickly as possible. Real world problems that stand to be affected include dispersal of supplies to areas damaged by a natural disaster, path planning by robot teams, and parcel delivery. Substantial improvements to existing methods could potentially benefit many lives.

6 References

References

[BF97] A. Blum and M. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90:281–300, 1997.

[BG00] B. Bonet and H. Geffner. Planning with incomplete information as heuristic search in belief space. In Steve Chien, Subbarao Kambhampati, and Craig A. Knoblock, editors, *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*, pages 52–61. AAAI Press, 2000.

[BK99] David Meir Blei and Leslie Pack Kaelbling. Shortest paths in a dynamic uncertain domain. In *Proceedings of the IJCAI Workshop on Adaptive Spatial Representations of Dynamic Environments*, 1999.

[BL99] Avrim Blum and John Langford. Probabilistic planning in the graphplan framework. In *the 5th European Conference on Planning (ECP-99)*. Springer-Verlag, 1999.

[CBGF97] A. Chowdhury, E. Burger, D. Grossman, and O. Frieder. Drs: A fault tolerant network routing protocol over ip. In *Proceedings of IEEE-IC3N Sixth International Conference On Computer Communications and Networks*, 1997.

[CR00] Alessandro Cimatti and Marco Roveri. Conformant Planning via Symbolic Model Checking. Technical Report 0006-04, ITC-IRST, Via Sommarive 18, 38055 Trento – Italy, June 2000. Submitted for publication to the Journal of Artificial Iintelligence Research.

[KLC95] L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. Technical Report CS-96-08, Brown University, Providence, RI, 1995.

[KNHD97] J. Koehler, B. Nebel, J. Hoffmann, and Y. Dimopoulos. Extending planning graphs to an adl subset. In *the 5th European Conference on Planning (ECP-97)*, pages 273–285. Springer-Verlag, 1997.

[LB93] M. Littman and J. Boyan. A distributed reinforcement learning scheme for network routing. In *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications*, pages 45–51, 1993.

[LF98] D. Long and M. Fox. Efficient implementation of the plan graph in stan. *Journal of AI Research*, 10:87–115, 1998.

[Lit95] M. Littman. Solving partially observable markov decision processes via vfa. In *Proceedings of the Workshop on Value Function Approximation, Machine Learning Conference 1995*, 1995.

[LST01] L.V. Lita, J. Schulte, and S. Thrun. A multi-agent system for agent coordination in uncertain environments. In *Proceedings of the 5th International Conference on Autonomous Agents*, 2001.

[Nil82] N. Nilson. *Principles of Artificial Intelligence.* Tioga Publishing Company, 1982.

[PR95] Ronald Parr and Stuart Russell. Approximating optimal policies for partially observable stochastic domains. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995.

[TFB00] S. Thrun, D. Fox, and W. Burgard. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000.