Abstraction for Planning Under Uncertainty

Lucian Vlad Lita LITA@CS.CMU.EDU
Computer Science Department, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213 USA

Sebastian Thrun THRUN@CS.CMU.EDU
Computer Science Department, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213 USA

Jamieson Schulte JSCHULTE@CS.CMU.EDU
Computer Science Department, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213 USA

Abstract

Mobile agent systems that operate in uncertain environments face many intractable issues that deal with planning and execution. Typical approaches rely on heuristic-based planners to provide reasonable solutions. However, in real life agents and control structures are bounded by computation power (i.e. memory, CPU frequency). When such systems are applied on larger environments they are not scalable.

We propose an *abstraction*-based approach independent of the planners being used, which brings flexibility and better scalability to mobile agent systems. Although it has not been thoroughly studied before, abstraction for planning in uncertain environments provides a way to specify a trade off between computation power and performance. Our results show that abstraction allows planners to limit their computation capabilities and still achieve good performance.

1. Introduction

Consider the problem of finding a shortest path to a goal location in a graph, where individual edges of the graph might or might not be traversable (BarNoy91). Whether or not an edge is traversable can only be observed by moving there. The only prior knowledge that is available consists of probabilistic statements about the traversability of all edges in the graph. This planning problem is known as the Canadian Traveler Problem (CTP).

1.1 Planning in Uncertain Environments

An optimal solution to the CTP can be viewed as a contingency plan which offers alternative routes if edges are found not to be traversable. The CTP differs from other hard planning and optimization problems in that the state of the environment is only partially observable. It can be viewed as a partially observable Markov Decision Process (POMDP) with local and deterministic observations (Kaelbling97, Littman98, Blei99). Finding an optimal contingency plan is known to be NP-hard (Blei99). In practice, due mostly to planning complexity, optimal solutions can only be found for very small graphs (e.g. 10 edges) and in multi-agent systems only for a small number of agents (e.g. 3). However, many practical CTP settings consist of large environments with hundreds of edges and hundreds of agents. The rest of this paper focuses on solving the scalability issues of single agent [1] system.

Real world solutions to the CTP need to factor in various limitations on computation. A classical time limitation example is packet routing in the Internet, where communication nodes might be up or down at random (Itai96). Clearly, packet routing has to be performed with extremely limited computation, making extensive planning infeasible. In other cases, agents planning in uncertain environments might not be bound by planning time, but might be constrained by planning depth since the number of potential future states is intractable. In Multi-robot exploration teams (Burgard00, Simmons00, Yamauchi98) of robots jointly explore an unknown environment. In this case he relative speed of the moving agents (robots) is much slower than in the packet routing problem, offering the oppor-

[1] Concurrently, promising research by Schulte et all is addressing the agent coordination and communication issues.

tunity for more extensive planning, but being limited by the amount of memory available.

Figure 1. During and after Hurricane Mitch, supply and rescue trucks need robust planning in an uncertain West Honduras

The main problem driving this research consists of natural disaster scenarios. In such settings agents (e.g., supply trucks) are tasked to bring emergency supplies to a number of population centers in a country devastated by a natural disaster. The road conditions are unknown. Hence, the optimal solution must offer backups from non-traversable roads. In particular, the natural disaster studied in this paper takes place in October 1998 in Honduras, while and after Hurricane Mitch (Figure 1). destroyed up to 90% of the country's infrastructure in certain regions.

1.2 Bounded Planning

Although, there exist planners that work well on CTP type problems (Lita01), they are very limited by their heuristics and by the amount of computation power available. Current solutions also vary widely depending on the agent initial and goal locations, as well as on the size and structure of the environment(e.g. graph). The farther apart the initial and goal locations are, the less robust will be the plan that the agent generates. This plan will account for few contingencies since the paths to the goal will be longer. Hence, the agent needs to put its resources into planning for a greater distance and cannot account for many contingencies. Shorter distances generate more robust plans while longer distances generate weaker plans, far from optimal.

In order to increase the flexibility and reach of existing planners, we propose to apply the method of abstraction in conjunction with planning under uncertainty. Most frequently used abstraction methods employ distance measures and probability measures to reduce graph complexity. By applying abstraction to probabilistic environments, agents can capture more complex notions such as expected time (or distance) or expected reward. Thus, abstract graphs are derived from the more complex base graphs using the measure that matters to the agent. Furthermore, we recognize the need for application specific abstraction methods and therefor we employ the use of various heuristics that guide the abstraction process. Abstraction provides the means to enhance the robustness of the plan or conversely to preserve the robustness even when stricter computation conditions are present.

2. Previous Work

Planning problems in deterministic settings have been widely studied. Techniques, heuristics, and planners have been developed and work quite well in polynomial time, generating solutions close to optimum (McDermott78, STRIPS). However these solutions do not work well when computation limitations are placed and when environments are very large.

Many researchers have also approached planning under uncertainty mostly from the more general framework of Markov Decision Processes (Littman98, Kaelbling97, Blythe99, Blai99). There are many finite horizon solutions to planning under uncertainty problems, especially POMDP style problems. However, the CTP is different since once a road condition is observed, it doesn't change over time. Also, in realistic environments such as the road structure of Honduras, the search horizon can be quite far for current solutions to yield near-optimal results.

This paper shows how to apply abstraction with an appropriate reward measure to planning in uncertain environment. However, most of the research done in abstraction is in deterministic scenarios. Abstraction in deterministic spaces has been studied in order to simplify huge deterministic graphs in the search for ways to generate fast plans (Valtorta87, Holte94, Sacerdoti74). Abstraction has also been addressed in planning problems, but more related to abstracting the plan space.

3. Approach

We developed the abstraction methods in conjunction to a multi-agent system that emulates the conditions of the roads in West Honduras during and after Hurricane Mitch passed through. We analyze single agents that are assigned random goals as well as starting positions. The agents generate plans in the Honduras environment and then execute them. In the planning process the mobile agents estimate a discounted reward $\widehat{R}$. The reward is based on the state s the agent might reach, and the observation history h that might have brought it to that s

$$\widehat{R}(s,h) = P(s|h) \cdot e^{-\alpha(t_o(s)+t_u(s))} \qquad (1)$$

where the reward depends on the probability $P(s|h)$ to reach state s given an observed history h, as well as the time it took to reach s and the estimated time it would take to reach the goal. α is a decay constant that tunes the reward to appropriate situations.

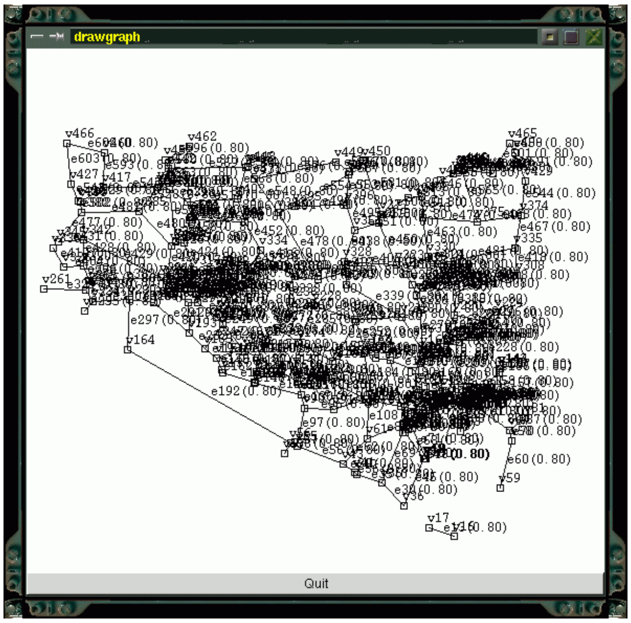

Figure 2. West Honduras' road structure with over 200 intersections (nodes). The BA* planner without strict computation limitations performs adequately in this environment.

3.1 Planning

Current planners are performing adequately most of the them given enough resources. Such planners could be used for CTP environments similar to Honduras. In this research we used the Belief A* planner (BA*), an efficient planner (Lita01) that works well in uncertain environments and works well on the disaster scenario we are considering. BA* is based on a A*-type heuristic (Nilsson82) that uses future discounted reward in order to search the plan space. Thus the search is done according to the reward measure described above.

Heuristic planners typically search the plan space and generate the best plan they can find. However, for large graphs, great distances, and high connectivity, they are forced to account for very few contingencies. This is especially true under computation power limitations.

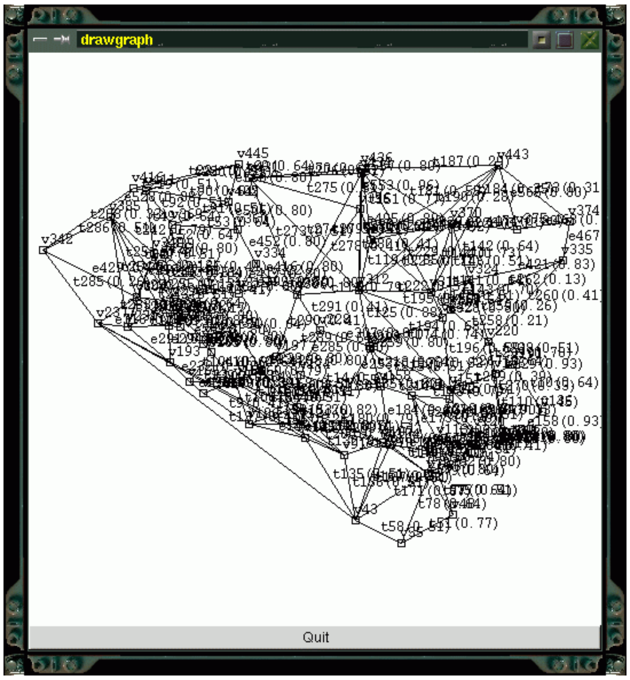

Figure 3. Abstract version of West Honduras' road structure with less than 100 intersections (nodes). Using the abstract step the BA* planner with stricter computation limitations can perform comparably.

3.2 Abstraction

We present abstraction as a flexible way to cope with known limitations of current planners. Abstraction allows planners to overcome these limitations and approach the performance they otherwise achieve if they had more resources.

During the abstraction process we eliminate nodes that are considered to have the least impact on the graph, but we preserve most of the information content of the paths that contained that node. Thus, we obtain a simpler abstract graph that can be more tractable for the limited planner. Figure 2 illustrates a complex and very dense base graph, while Figure 3 shows its abstract, more tractable version. Nodes are removed from the graph according to heuristics that are the most appropriate for specific environments.

Choosing among several layers of abstractions provides the agent with the a computation/performance trade off. Our abstraction method aims to preserve the reward the agent would obtain in any abstract graph. The goal is to be able to dictate an abstract course of action that yields a reward very close to the reward in the complex base graph. Since the abstraction method is not based on distance or probability solely, the abstract graphs are derived from the base graph through the reward measure relevant to the agent.

3.2.1 Abstraction Heuristics

As an initial step in the abstraction process deterministic abstraction can be performed. This step simplifies the graph while not affecting performance in a negative way. Nodes with degree zero, and one can be removed without loss of plan quality. Nodes with degree of two can be removed and their adjacent nodes connected instead. This replaces long stretches of roads with one equivalent road. Also, tours graph components that are connected to any paths through only one node, if explored do not gain any information for the agent since they go back to the same unique connecting point. Thus they can be removed without loss of useful information.

During the initial abstraction step described in Table 1, a critical step is choosing the node to be removed from the graph. Thus a heuristic must be used in determining a node ordering. This ordering must reflect the agent's optimization measure, as well as the particular graph structure. In this research we used various heuristics in order to observe performance variations with different abstraction heuristics.

A random heuristics insured that nodes would be sampled from every part of the graph. This particular approach assumes that the graph is structurally self similar and probabilistically self similar (i.e. the Hurricane damaged the all parts of road structure in a similar fashion).

Two other heuristics that reflect the graph's connectivity were used. Both heuristics measure the connectivity degree of the nodes in the base graph. One of the heuristics selects the node with the highest connectivity degree node to be removed, while the other performs the opposite selecting the lowest connectivity degree node to be removed.

Another heuristic class can be based on distance measures. If a node is very close to a high degree node from an Euclidean perspective, it could be removed. Clustering based on Euclidean distance is also a feasible basis for a heuristic.

Also probability based heuristics can also make a difference. Heuristics that remove nodes that are very probabilistically close to high degree nodes could be removed. Also, probabilistic clustering could produce various other heuristics.

Other heuristics could and should be used as well as long as they incorporate useful prior knowledge about the environment and agent's optimization measure.

3.2.2 Abstraction History

During the abstraction process, nodes are removed and edges connected to them are merged into abstract edges in order to preserve the information content of paths trough those nodes. In the merging process, we need to maintain the knowledge of how the abstract edges were formed, so that it is easy to compute abstraction edges probability and expected length. We used Binary Decision Diagrams (BDDs) which capture both the edge dependencies and the type of merging that was performed: sequential or parallel (Table 1).

3.3 Execution

During the execution step, the agents follow the abstract plan attempting to find the best available path through the abstract nodes to the goal. Locally, in the base level, the agent generates a simple plan using non-abstract nodes to reach the following step in the abstract plan. Local planning can consists of planning in a very small space for the next sub-goal: an abstract node in the abstract plan. The BDD computed in the abstraction process carries information that can be used to approximate the expected reward and guide the local planning.

Since the abstraction process is based on a relevant heuristic and incorporates merged paths from previous abstraction levels, the expected reward computed throughout planning and execution (agents replan after each move) is close to the actual reward that would have been generated by the base graph.

4. Evaluation

The simulations were performed in a multi-agent system simulating West Honduras during and after Hurricane Mitch. The graph representations of parts of West Honduras consist of more than 200 nodes and 250 edges. We varied the realistic scenarios through various realistic instances in during the disaster. We also varied the abstraction heuristics as well as the decay parameter responsible with regulating the reward over time.

We measured performance of different abstraction lev-

Abstraction Algorithm
While abstraction level not reached perform the following steps:

1. According to a heuristic, *select* the least important *node* **X**. It could be a node close to a high degree node or nodes that do not belong to high degree paths.
2. *Remove selected node* **X**from graph and
   (a) Preserve the most important paths throught **X** by apropriately connecting **X**'s neighbors.
      i. Replace every edges E_1 and E_2 connected to **X** with an abstract edge E_S that connects the two neighbors of **X**. Due to a *sequential merging*, E_S has the following properties:
         A. Prob (E_S) = Prob(E_1) * Prob(E_2)
         B. Dist(E_S) = Dist(E_1) + Dist(E_2)
      ii. If the two neighbors were previously connected with an edge E_3, *Merge* E_S and E_3 into a new abstract edge E_P in a *parallel* fashion:
         A. Prob (E_P) = Prob(E_S) + Prob(E_3) - Prob(E_S)*Prob(E_3)
         B. Dist(E_P) = Prob(E_S)*Dist(E_S) + [1-Prob(E_S)]*Prob(E_3)*[Dist(E_S)+Dist(E_3)]
   (b) Using a *Boolean Decison Diagram* (BDD), a directed acyclic structure that keeps track of how each abstract edge was formed. Trough the BDD we maintain a *list of dependencies* between arcs in the graph. This way given any abstract edge, we can track its ancestors and we can recreate the formula (a series of sequential and parallel merges) that was used to compose it.

Table 1. The abstraction algorithm is performed before in order to provide a simpler graph for planning. The BDD is used to help make the transition back to the base graph during short distance planning.

els under increasingly stricter computation restrictions. We compare the abstraction performance with the baseline planner using no abstraction, and loose computation restrictions. Performance was measured through the time the agent takes to reach the goal or equivalently the distance the agent travels to reach the goal. For the BA* computation restrictions were imposed through limiting its search capacity: the number of states in the plan space it could evaluate.

5. Results

Experiments were performed on large West Honduras graphs (more than 200 nodes and more than 250 edges) using different realistic scenarios reflecting the Hurricane Mitch disaster in Honduras. Probabilistic knowledge about edges varied from scenario to scenario, but were always within the interval 0.2 and 0.8. Some scenarios considered uniform probability distributions (the hurricane destroyed roads with similar probability) and others with destruction localized to areas of the graph.

We used 5 levels of abstraction and we compared performance achieved on those levels with the performance of the baseline planner (with no abstraction).

Figure 4 shows the performance of the BA* planner on a part of West Honduras. The first datapoint is the baseline and represents a graph with 234 nodes and 300 edges. Subsequent abstraction levels represent graphs with 210 nodes and 273 edges, 123 nodes and 186 edges, 111 nodes and 189 edges, 99 nodes and 195 edges, and 82 nodes and 196 edges respectively. The reward is regulated by a decay constant of 0.00008. The BA* planner limitations corresponding to each abstraction level is increasingly stricter. The baseline planner with no abstraction has an evaluation capacity of 50000 states in the search space. Subsequent abstraction level have 30000, 20000, 10000, 5000 and 1000 state exploration capacity.

The experiments generate similar results for different time decaying rewards, and for different scenarios. Figure 5 shows a similar experiment involving the same computation limitations and with a different reward constant of 0.0002. The computation bounds and the structure of the graph are kept the same.

Up to the third abstraction level, the performance is comparable with previous levels with more computation power. However the subsequent abstraction levels show a drastically decreasing performance. This suggests that if the abstraction is too deep it eliminates nodes that best capture the nature of the graph. Also, starting with the fourth abstraction level, the number

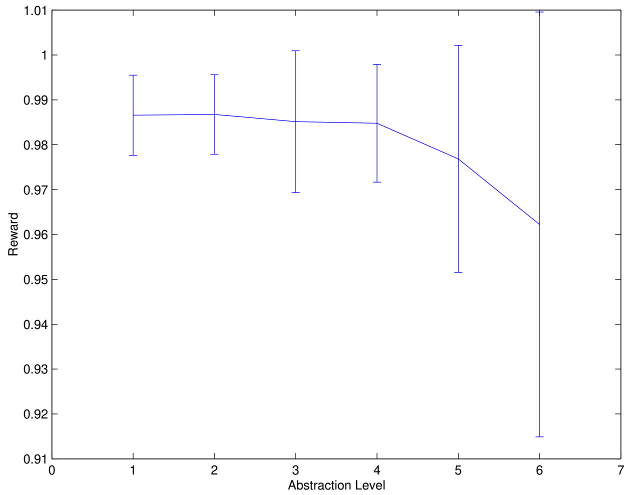

Figure 4. Reward obtained by agent in a West Honduras simulation. The first datapoint is the baseline, and 5 abstraction levels follow. The decay constant is 0.00008.

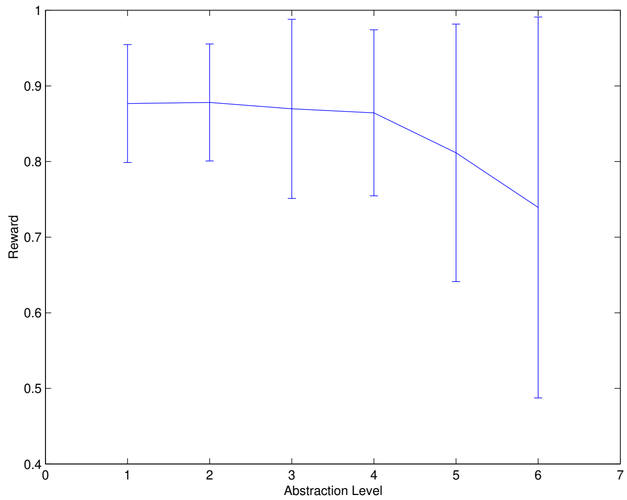

Figure 5. Reward obtained by agent in a West Honduras simulation. The first datapoint is the baseline, and 5 abstraction levels follow. The decay constant is 0.0002.

of edges starts increasing again due to the increasing connectivity needed to preserve high expected reward paths. This makes it harder for the planner which has to traverse many of these potentially beneficial edges, especially since computation power decreases with abstraction level. And finally, since full connectivity cannot be preserved when merging edges, potentially useful information is lost when even more nodes are removed. This suggests in this part of West Honduras, a good choice given performance and computation power would be the third abstraction level.

These results show that by using this abstraction method, planners that are limited by computation power aspects can scale further to achieve comparable performance in larger environments. The findings also show that agents can trade off computation power for performance. When using abstraction, performance decrease can be kept minimal offering the agent the flexibility in when operating in uncertain environments.

5.1 Limitations

Abstraction does not always perform well with planners. Table 2 shows three very different results obtained on the same West Honduras scenario. The first agent cannot reach the goal with its given restrictions on computation power. However, with the third level of abstraction, the agent can plan to the goal. The second agent is provide for comparison purposes and is a representative result on West Honduras. The performance versus abstraction curve is similar to results such as in Figure 2. The third column suggests that abstraction was not suitable for this particular agent and it made the planning process more complex than it was without abstraction. However our experiments found that the first and third agents are extremes cases in our disaster relief scenarios. Other graphs and probability distributions might make it detrimental to use abstraction when planning.

Table 2. Extreme cases surface when abstraction is used with planners under uncertainty. In the West Honduras experiments extremes such Agent 1 and Agent 3 are very rare.

Abs. Lvl	Agent 1	Agent 2	Agent 3
Baseline	0	0.984305	0.953444
Level 1	0	0.984305	0.954206
Level 2	0.940385	0.984305	0.878024
Level 3	0.937167	0.98430	0.946498
Level 4	0.96253	0.909333	0.658588
Level 5	0.928165	0.742515	0

6. Discussion

Using the Hurricane Mitch developed an abstraction method to be used in conjunction with planning in uncertain environments. Our experiments show that in a wide range of situations abstraction provides a way for computationally limited planners to extend their reach to larger environments and perform close to planners with more resources. This method allows the agent to choose the abstraction level that most appropriately reflects the trade off between performance and computation restrictions.

The results show that using the abstraction method presented in this paper together with the BA* planner produces promising results. Three levels of abstraction usually provide the best tradeoff when looking at the specific computation limitations used in our experiments, but the actual computation/performance trade off can be at the agent's discretion.

6.1 Potential Future Work

Further work needs to be done in bounding the local planning steps in order to make it more straight forward to change abstraction levels. Further scalability might also emerge from better path merging in the abstraction process.

It would be worthwhile applying the abstraction method in an environment which involves multiple agents attempting to reach multiple target locations at the same time. Instead of communicating information based on the original complex graph, agents could communicate less information based on an abstraction level. The abstraction level does not need to be the level used internally by the agent.

Finally, more heuristics could improve the performance, making use of prior knowledge about graphs and probability distribution/events. Further work could apply this method to other real world systems as well.

References

Blei, D. M. and Kaelbling, L. P. (1999).*Shortest Paths in a Dynamic Uncertain Domain*, IJCAI Workshop on Adaptive Spatial Representations of Dynamic Environments.

Burgard, W. and Fox, D. and Moors, M. and Simmons, R. and Thrun, S. (2000). *Collaborative Multi-Robot Exploration*, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)

Yamauchi, B. and Langley, P. and Schultz, A. C. and Grefenstette, J. and Adams, W. (1998). *Magellan: An Integrated Adaptive Architecture for Mobile Robots*, Institute for the Study of Learning and Expertise (ISLE)

Simmons, R. and Apfelbaum, D. and Burgard, W. and Fox, D. and Moors, M. and Thrun, S. and Younes, H. (2000). *Coordination for Multi-Robot Exploration and Mapping*, Proceedings of the AAAI National Conference on Artificial Intelligence

Matari, M. J. (1997). *Reinforcement Learning in the Multi-Robot Domain*, Autonomous Robots, 4, pg 73-83

Nilsson, N. J. (1982). *Principles of Artificial Intelligence*, Springer Publisher, NY

Bar-Noy, A. and Schieber, B. (1991). *The Canadian Traveler Problem*, Proceedings of the 2nd ACM-SIAM Symposium on Discrete Algorithms.

Itai, A. and Shachnai, H. (1996). *Adaptive Source Routing in High-Speed Networks*, Journal of Algorithms, 20, pg 218-243

Kaelbling, L. P. and Littman, M. L. and Cassandra, A. R. (1998). *Planning and Acting in Partially Observable Stochastic Domains*, Artificial Intelligence, 101,1-2, pg 99-134

Blythe J. (1999). *An Overview of Planning Under Uncertainty*

Tash J. K. (1993). *A Framework for Planning under Uncertainty.* AAAI Spring Symposium pp. 137-141

McDermott D. (1978). *Planning and Acting.* Cognitive Science, 2, 1978.

Holte R. (1994). *Searching with abstractions: A unifying framework and new high-performance algorithm.* In Proceedings of the Canadian Conference on Artificial Intelligence, 263-270.

Lita L. V., Schulte J., and Thrun S. (2001). *A Multi-Agent System for Agent Coordination in Uncertain Environments.* Poster to appear at the 5th International Conference on Autonomous Agents.

Schulte J. (2001). *Probabilistic, Distributed, Multi-Agent Planning in Uncertain Environments*