Improving the Performance of Planning Systems Using Parallel Hardware and Flexible Social Laws

Diane J. Cook

University of Texas at Arlington
cook@centauri.uta.edu

Abstract

The objective of this research project is to improve the applicability of artificial intelligence planning systems to large-scale tasks. This paper details two ongoing aspects of this work. First, we describe a parallel/distributed approach to speeding up the heuristic search that drives machine planning. Second, we introduce a mechanism called *flexible social laws* that reduces the amount of computation and communication necessary to coordinate multiagent plans.

One application of artificial intelligence techniques that captivates researchers is the development of automated planning techniques. A semi-automated or fully-automated agent can be used to perform tasks that are too tedious, hazardous, or sometimes complex for humans. Although a large variety of tasks can be assumed by automated mechanisms, the time required to compute plans prohibits widespread use of planning techniques. Even the problem of computing a plan to move a multi-jointed robot arm from one position to another can be computation-intensive, yet manufacturing tasks add several degrees of complexity to this problem. The problem is further exacerbated when plans must be generated an coordinated for multiple agents.

We are currently researching methods of improving AI planning systems by developing parallel search techniques to speed up the task and by using flexible social laws to reduce communication and computation inherent in multiagent systems. Parallel processing can considerably reduce time spent in search, and thereby speedup many AI techniques including planning. We parallelize IDA* search because this is an admissible search algorithm that requires only linear memory. IDA* consists of a series of depth-first searches, in which each pass through the space searches up to a incrementally-increasing cost limit.

A number of techniques exist that make use of parallel hardware to improve the performance of heuristic search. Although many reported approaches can offer benefits for a subset of problem types, no benefit reports consistent speedup. In addition, many of the proposed approaches are complementary to each other.

We are designing a hybrid system, EUREKA, that integrates a number of the features of existing approaches to parallel and distributed search. EUREKA makes use of the C4.5 machine learning system to create a rule base of parallel/distributed search strategies. When a new problem domain is encountered, EUREKA searches the space to a small depth to obtain information about the tree, then consults the learned rule base to select a strategy for the remainder of the search.

EUREKA offers the following search strategies:

Task Distribution. Some parallel search techniques parallelize the redundant iterations of IDA* (parallel window search) while other techniques distribute the search subtrees among available processors (distributed tree search). *Eureka* actually combines the two approaches by selecting

a cluster size. Each cluster is assigned a unique cost limit (parallel window search) and distributes the tree among processors within the cluster (distributed tree search).

Tree Ordering. Because IDA* searches the left subtree completely before moving to the next subtree, the left-to-right ordering of the tree can greatly improve performance. *Eureka* can choose to order children of a particular node or actually reorder the choice of operators between iterations using information gained about the tree from the previous iteration.

Load Balancing. To prevent processors from idling, *Eureka* can allows processors to request work from other processors when they become idle. The load balancing technique can be refined to adjust the amount of work shared between processors and to determine which processor to approach for work.

Features of the tree that can influence the choice of search strategy include the average branching factor, the heuristic branching factor, the imbalance of the tree, skewness of the tree, heuristic error, type of machine architecture and number of available processors, preference of optimal or near-optimal solution, and position of favorable nodes in the tree. To date, *Eureka* has been shown to outperform any strategy used in isolation on instances of the Fifteen Puzzle problem, the robot arm motion planning problem, and an artificially-generated search space.

A second ongoing project introduces a method by which agents may reduce both planning and communication costs by planning with stringent social laws, relaxing the laws as needed to find a solution. Consider three mobile robots negotiating over a three-lane road. If we are to prevent collisions, agents must be sure in advance that their planned paths will not bring them into the same sections of the highway. In the absence of any conventions for limiting possible locations of the agents, to avoid collisions each agent must inform all the others of every step of its plan. If there is a conflict, one or more of the agents will have to change their plans. This results in exponential computation and communication costs.

The *social law* represents a method of limiting the agents' options to reduce both plan generation cost and costs from interaction in cooperative or uncooperative environments. One possible mobile robot social law is "drive on the right". This law will prune a portion of the search space, since agents will not need to consider moving into other lanes. It will also reduce possible contention, since agents moving in one direction will never share a lane with those moving in the opposite direction. Unfortunately, social laws can prevent us from finding a solution. If one lane has a roadblock, the agent required to move in that lane cannot find a solution.

When a multiagent system uses *flexible social laws*, agents prefer to obey the "laws" but are able to relax them. Assume a ranking of sets of social laws, from strictest to most lenient, where a more lenient set is one that allows a greater choice of operators. Also assume a limit on the depth of search. Each agent will try to generate a plan within the strictest set of laws. If this fails, the agent will try again, using the next set in the ranking. This continues until the agent has a successful plan or has failed with the most lenient set of laws.

Flexible social laws can reduce both computation and communication. Soundness is not sacrificed, because if no solution is found within limits set, the agent plans with more flexible laws. Of course, when an agent finds a solution while using a social law, the solution may not be optimal. However, the cost of plan generation may be high enough that a solution quickly found is preferred to an optimal solution.

In our ongoing research, we have designed an implemented a system that both learns and enforces flexible social laws. This system has been tested in a mobile robot domain and an Internet agent domain. We have theoretically shown the potential benefits and costs of this approach.