CS 396 Special Topics in Artificial Intelligence
Project Deliverable 3

Ari Wilson

February 27th, 2007

Progress

I have made significant progress in developing the basic structure of the bidding scenario presented in my project description. There are three fundamental structures within which agents may be given priorities, strategize, learn, collect resources, or perform bidding or auctions. These structures are:

- *Meta-game*: Before a game begins, each agent is given randomized priorities, p_1, p_2, $\ldots$, p_k, for each of k resources, designated r_1, r_2, $\ldots$, r_k. To maximize granularity, these priorities are represented as floating point values between 0 and 1, inclusive, where 1 is the highest priority possible.

- *Game*: A total of g games are played as follows. During each game, each agent is given a randomized starting position and a constant amount a of each discrete resource is divided up and placed randomly in the agent environment, which is assumed to be fully observable. The number of such *resource piles* of each resource will be strictly less than n, the number of agents in the game, so that it is impossible for all agents to have some of a type of resources in the game.

- *Round*: Agents participate in two rounds in each game. The first round involves *resource collection*, where all agents attempt to gather resources within the environment by moving to and collecting some resource. Movement speed is assumed to be identical between agents, and the time it takes to collect a resource pile is proportional to the amount present at that location. In this round, agents can observe how much of each resource any other agent collects at the expense of available movement and collection time. This round ends once all resource piles have been collected. In the second round, all agents are allowed to *advertise* their resources, *auction* their resources, *inquire* about resource availability, and *bid* on other agents' resources. This round ends when no agent believes it can further increase its individual or group utility by creating a new auction.

No agent is currently allowed to plan, sense, communicate or act except during the two rounds of each game. After each game, statistics are collected on ranking agents' individual utility and group utility, as well as on overall group utility, calculated as the sum of each agents' individual utility. The group utility is given for each individual as the utility an agent creates for the group as a whole by selling and buying resources. After a given meta-game (encompassing a large number of individual games), these ratings are averaged and each agent is given an individual rating, a contribution rating, and the group of agents as a whole is given an overall rating.

This scenario is developed as a simplified model of multi-agent discovery cooperation and competition in a resource-constrained environment. For example, robotic platforms attempting to build a lunar colony on the dark side of the moon may encounter limited resources in the form of water, certain metals, and gases. All of these materials may be needed in certain measure by agents tasked with separate operations with (presumed) equal priority. Agents will be required to collect these resources in an expeditious fashion and distribute them in an appropriate fashion so that each agent can complete its task to the best of its ability and resources. Agents may or may not have complete *a priori* knowledge of each others' priorities. In a competitive environment, software agents may have to find, share, and utilize limited processing time, memory, and other resources in host networked systems. This scenario can be contrasted with the Travel Agent scenario developed in the International Trading Agent Competition[1] in order to delineate my focus on aspects of learning rather than the other facets of strategy and agent behavior.

At this point, there are only two simple agents that are capable of playing the game, making the results mostly uninteresting, as the agents will just trade the items they need to each other unless there is a conflict of interest, in which case they will hoard the items they need or attempt to deny the other agent the resources it needs. The two existing agents are designated `BasicGreedyAgent` and `BasicHoardingAgent`.

`BasicGreedyAgent` is a simple agent that attempts to maximize its utility in a given game without considering anything about the other agents operating in its environment. It gathers resources by examining all resource piles within a given distance d, sorting these resources by the utility they give minus the amount of time needed to move to that location and process that pile. In the bidding round, it operates by checking the current resources and quantities being sold against the resources and quantities in its inventory. It bids for these items in order of the utility that would be added were it to possess the additional quantity of resource. This utility is calculated as $p_i q_i$ where i is the resource being bid on and q_i is the quantity of i for sale. The agent will randomly choose an item worth less than a for sale item to bid, in equal quantity if possible, less if not. If it has no item with priority less than the item up for sale, it will not place a bid. It will advertise an auction for an item in its possession randomly if its priority is less than some small fixed value and accept bids for quantities of items that will increase its overall utility.

`BasicHoardingAgent` operates on the principle that a bird in the hand is worth two in the bush. In most cases, it will operate similarly to `BasicGreedyAgent`. However, when deciding whether or not to move to a location to collect a resource, it will weight the distance to a resource pile as a more heavily negative factor by some constant scale factor. The reasoning behind this decision is that another agent probably wants the resource, and if it is very far away, it is more likely to get it than our agent. As well, although it may place bids on items it sees as increasing its utility, it will not create its own auctions on the principle that even its lesser valued resources are probably very valuable to other agents.

More agents with differing, more general strategies are being developed. As well, the existing agents are being modularized so various parameters important to their strategy can be changed. This will allow me to easily introduce new agents to the game, if the changes made are significant, or modify existing agents, if the changes made are minor. In this way, agents may eventually be tuned by the meta-game controller itself based on the success or failure statistics of the unmodified agents. These are all examples of reinforcement learning at the meta-game level - learning how to play the game better by reflecting upon the outcomes of individual trials. At the moment, however, I am focusing on learning on a lower level, between games. That is, I want each agent to attempt to discover the priorities of all other agents along with some notion of their strategies.

Schedule

Although I have advanced my project's implementation and theory a fair amount in the last week, the goal of being to the point where each agent can reason independently in a game-theoretic or Bayesian manner remains distant. As I have chosen to focus on creating simple agents that implement more complex learning methods, this portion of my original project proposal may be mostly dropped or investigated at a later time.

Knowledge and beliefs

Knowledge and beliefs are fundamental to my agents. Knowledge about an an agent's own resource supply and those of others forms a basic component of its decision on whether or not to bid on a particular auction or create an auction of its own. An agent's beliefs on the strategies and priorities of other agents also help to inform these decisions. Both of these information structures are required for an agent to act in a more intelligent fashion than the two agents I have outlined above and their accuracy and completeness is essential for an agent to compete or cooperate successfully in the dynamic environment that forms the cornerstone of my project.

However, the data structure representations of these objects do not have to be very complex in order to represent the environment effectively. The knowledge of an agent can be effectively represented as an matrix representing agents and quantities of resources, updated after each observed resource collection operation or completed auction. The beliefs of an agent about another agent are used to form a best response to the other agent's placed auctions or bids by either bidding or not and accepting bids or not. Beliefs on other agents' priorities can be expressed, again, as a matrix of agents and hypothesized priorities for each resource, along with confidence values for each hypothesis, updated based on actions or inactions taken by an agent in pursuit of a resource over the course of several games. These representations are sufficient for my overall project goals by allowing me to test methods of learning by seeing how they affect the accuracy of these beliefs. Understanding another agent's overall strategy, on the other hand, is a highly non-trivial matter in anything more complicated than tic-tac-toe and will probably turn out to be outside the scope of this project. Even in the extremely simple game of Rock Paper Scissors, the very best agents turn out to use extremely complicated methods to predict the strategies of others[1]!

In summary, the accuracy and completeness of agents' beliefs and knowledge in my simulation environment will be the deciding factor in how they fare in both maximizing their own utility and that of the group as a whole.

[1] Seen at the RoShamBo programming competition, which is described at `http://www.cs.ualberta.ca/~darse/rsbpc.html`

References

[1] Amy Greenwald and Justin Boyan. Bidding algorithms for simultaneous auctions: A case study. *Autonomous Agents and Multi-Agent Systems*, 10:67–89, 2005.