SHABaN Multi-Agent Team To Herd Cows

Adel T. Rahmani, Alireza Saberi, Mehdi Mohammadi, Amin Nikanjam, Ehsan Adeli Mosabbeb, Monireh Abdoos

Iran University of Science and Technology
{*rahmani*, *a_saberi*, *mh_mohammadi*, *nikanjam*, *eadeli*, *abdoos*}@*iust.ac.ir*

Abstract. This paper is submitted as the final team description of SHABaN[1] team, one of the participants in the Second Multi-Agent Programming Contest in association with the ProMAS 2008 workshop. Here we describe the agent architecture and behaviors to solve a cooperative task in a highly dynamic environment. Our approach consists of evaluating strategies in NetLogo and a raw implementation.

1 Introduction

Multi-agent systems are composed of a number of interacting computing elements, also known as agents. Agents have two important capabilities: the abilities to take autonomous actions and interact with other agents [1].

Agent contest is an attempt to motivate research in the area of multi-agent system development and programming. The scenario this year is about cows and herders. Each team owns six agents, whose duties are to collect cows and guide them to the corral. To this end, we propose a multi-agent system to compete against the opponent through a sequence of rationale actions aiming the cooperation and coordination concepts.

NetLogo [2], a cross-platform multi-agent programmable modeling environment, is used for the simple prototyping and simulation of the strategies. After designing and testing the strategies, a programming language is employed to implement them.

The paper is organized as what follows: next section provides the information on the analysis and design of system. Section 3 illustrates the software architecture of the proposed multi-agent system. In Section 4 we briefly describe the team strategy and some algorithms. A discussion about the Contest is mentioned in section 5. Finally the conclusion remarks could be found in section 6.

2 System Analysis and Design

The SHABaN team is designed as 6 independent agents that communicate with a coordinator. Figure 1 shows the single agent architecture, coordination and

[1] SHABaN is a Farsi name that means shepherd

monitoring modules. The Communication component connects to the contest simulation server, parses the received information and passes them to the perceptor. The communication component sends back the desired actions and other necessary information to the server.

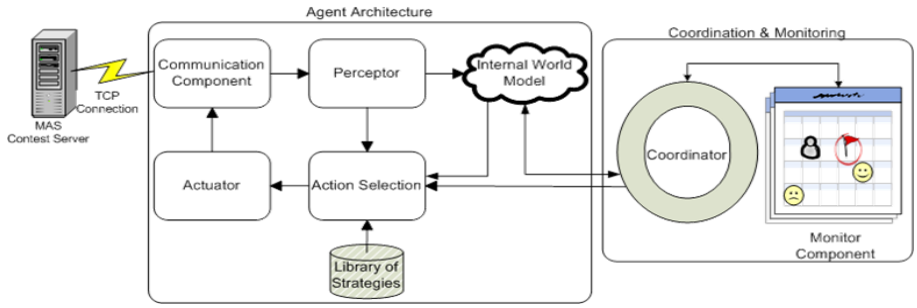

Fig. 1. Agent architecture, coordination and monitoring modules

The perceptor component senses the environment via the received information, and updates the internal world model. Action selection component uses the internal world model, preceptor, coordinator and library of strategies to select the best action. Actually there are two basic roles for agents: exploration and herding. In the exploration mode agents are to explore the environment but in the herding mode they target a cow and try to move it toward the own corral. Actuator prepares the appropriate command due to the selected action and passes it to the communication component.

The coordinator makes the comprehensive world model by using the internal world model of the all agents and a modified BFS algorithm. This comprehensive world model assists agents to find the best path to the corral and the nearest cow. The monitor component uses coordinator information to present a world view during run-time. The main usage of the monitor component is to ease the debugging process.

A high-level algorithmic description of SHABaN is shown in figure 2.

3 Software Architecture

This section gives an overview of the software tools used for simulation and implementation of SHABaN team.

3.1 Simulation

NetLogo is a multi-agent programming language and modeling environment for simulating natural and social phenomena. It was authored by Uri Wilensky in

```
Start Tournament:
  for each agent in team:
    Connect to simulation server using one TCP connection.
    Identify and authenticate yourself and receive acknowledgement
from the server and wait for the match to start.
    Start Match:
      Get match information
      Repeat for a finite number of steps:
        Each agent gets sensory information.
        Process the received sensory information.
        Update the internal world model.
        Each agent negotiates with the coordinator and selects an action
        or autonomously select an action based on its goal.
      Encode and send the actions to the simulation server.
      Once a while, analyze the situation and update the strategy.
```

Fig. 2. Algorithmic description of SHABaN multi-agent team

1999 and still is under development at the Center for Connected Learning and Computer-Based Modeling [2].

We use NetLogo to implement and evaluate our ideas before detailed implementation. Its simplicity lets us to test our strategies in a multi-agent environment quickly. Furthermore, while contest server was not available NetLogo was utilized as an appropriate simulation environment. Figure 3 shows a sample simulated environment using NetLogo.

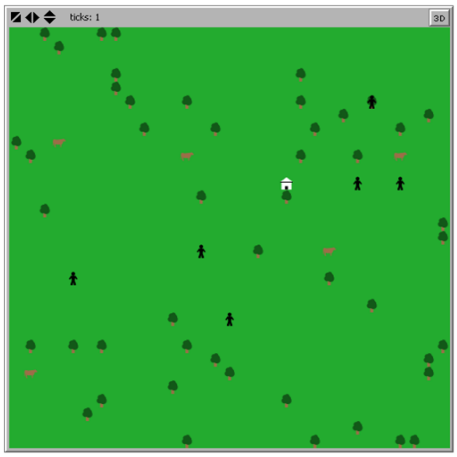

Fig. 3. A sample simulated environment using NetLogo.

3.2 Implementation

For implementation a multi-agent and a non-multi-agent based approach were firstly chosen. As the multi-agent approach, JIAC IV [3] was used. JIAC IV is a framework for development of powerful emergent intelligence and autonomous multi-agent systems. It is built on the basis of compact component architecture and uses a specific agent programming language, JADL [4].

Due to the lack of enough experience with JIAC and the short time before the contest, the JIAC IV approach was set aside after around a month of working with. A simple non-multi-agent based framework was implemented using a Windows platform and the C# programming language. All the algorithms and interfacing protocols were implemented using this language. But for long-term research, such a multi-agent implementation environment like JIAC, which proved to be efficient, is strongly advised.

4 Agent team strategy

We have defined two different roles for agents: exploration and herding. In exploration mode, agents travel to unknown or previously known areas and try to detect the environment. Once the agent decides to bring a cow home, the agent switches to herding mode. In the both modes once the agent receive the sensory information, it updates the internal world model and consequently the coordinator.

At the beginning of each match the grid is divided into some areas. Then agents in the exploration mode go to the nearest area assigned by the coordinator to explore that area. At each time step agents transfer their percept to the internal world model. After some time steps and exploring sufficient areas required for constructing the comprehensive world model, the coordinator makes agents to switch to herding mode by assigning a cow to herd. Then each agent autonomously tries to herd the specific cow toward the corral using a simple algorithm based on the distance of neighbor cells to the corral.

As it was mentioned before a modified BFS algorithm is used to construct the comprehensive world model. This model contains all grid information based on agent perceptions: cows, trees, corrals and other agents. The BFS algorithm assigns a semi-shortest path for each pair of cells in the environment. It uses an incremental algorithm, such that once a new aspect of the environment is discovered it updates all the previously found paths which could be improved using these newly discovered cells. The comprehensive model is constructed in coordinator based on all agents' perceptions and is reachable by all agents.

In maze-like scenario the blocking strategy can play a leading role, if a bottleneck is available and it can be detected by the agents. The coordinator finds narrowest part of the way to opponent corral as bottleneck and assigns at most two agents to patrol this area. This strategy will be effective if other agents do their best and move cows to own corral efficiently. In order to detect bottlenecks, the cells are divided in to groups based on their distance to opponent

corral and the group with smaller members has the chance to be a bottlenecks. Furthermore, it is not necessary to explore the entire map or the opponent corral neighborhoods to detect the opponent bottleneck in symmetric maps. We may find our bottleneck by exploring our corral neighborhoods and estimate the opponent bottleneck location by considering the map symmetry.

5 Discussion

We think that our first attendance in the Agent Contest was successful. Engaging in a time-consuming sophisticated process of developing multi-agent team is very useful as well as nice. We touch all the theoretic challenges in practice and discover our shortcomings. From our point of view the main critic is lack of effective cooperation. Individual behavior of SHABaN agents is acceptable for us but they can not cooperate effectively with each other in exploring the environment and herding.

Although Agents Contest 2008 is successful to serve its purpose to provide a testbed for multi-agent system programming but we argue some suggestions. It will be better if all teams come to a specific place to participate in the contest like RoboCup series. Face to face discussions and observing the design and execution of other teams will be really helpful. With remote online participation, we prefer that all matches in a specific grid take place consecutively and at one session. This approach seems to be much better than what is done in Agent Contest 2008.

6 Conclusion

In this paper we have demonstrated a multi-agent team, including six herders and a coordinator, aiming the goal of collecting cows as much as possible. SHABaN team strategies are first simulated and evaluated using NetLogo. It could usefully help us to extract efficient ideas. The first attempt to participate in this contest, for sure, has been a great experience and will ameliorate us for probable future events. We look forward to participating in Agent Contest 2009.

References

1. Wooldridge, M.: An Introduction to Multiagent Systems, John Wiley & Sons LTD (2002).
2. Wilensky, U.: NetLogo 4.0.2, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. (2007). http://ccl.northwestern.edu/netlogo.
3. Sesseler, R.: A modular architecture for service based interactions between Agents, PhD thesis, Technische Universitt Berlin (2002).
4. Konnerth, T., Hirsch, B., Albayrak S.: JADL-An Agent Description Language for Smart Agents. In Baldoni, M., Endriss, U., eds.: Declarative Agent Languages and Technologies IV, Volume 4327 of LNCS, Springer Verlag (2006) 141-155.