Hierarchical Controller for Robotic Soccer

Byron Knoll
Cognitive Systems 402
April 13, 2008

ABSTRACT

RoboCup is an initiative aimed at advancing Artificial Intelligence (AI) and robotics research. This paper first provides an overview of a robotic soccer competition hosted by RoboCup called the Small Size League. It then documents the design and implementation of an AI framework used by a team at the University of British Columbia which is aiming to compete in the Small Size League. Finally, it evaluates the effectiveness of the AI framework.

1. INTRODUCTION

This paper describes the design and implementation of an Artificial Intelligence (AI) framework made by a team which is aiming to compete in the Small Size League at the RoboCup soccer competition. This is an interesting domain for research not only because of the variety of unique approaches and techniques that can be used but also because of the challenging problems involved. Successfully overcoming these challenges could benefit many practical applications outside of the robotic soccer domain. RoboCup soccer provides an environment for testing AI in which many different techniques can be competed against each other in a structured competition. The entertaining nature of the competition also provides motivation for a large number of teams to invest the time and effort into creating effective playing strategies.

1.1 Background

Robotic soccer was first proposed by Alan Mackworth in 1993 as a testbed for research [1]. The first RoboCup competition took place in 1997. Since then there has been a substantial body of AI research related to robotic soccer. The official goal of the RoboCup competition is "by the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team."* There have been a large variety of unique approaches towards creating effective robotic soccer AI. A successful system was developed by Yu Zhang and Alan Mackworth that used Constraint Nets and an evolutionary algorithm to adjust parameter weights [2]. Research by Anthony Lee used genetic algorithms to test an alternative type of fitness function [3]. Many teams rely on programming skill and human intuition to hardcode playing strategies into the AI.

1.2 Small Size League

The Small Size League is one of a variety of competitions hosted by RoboCup. There are leagues for other tasks besides soccer (such as for search and rescue missions) and the competitions can either involve physical robots or just simulations. The Small Size League consists of robots no larger than 18cm in diameter. There are five robots on a team playing on a field slightly bigger than a ping-pong table. The soccer ball is the size of a golf ball. A match consists of two 10-minute halves.

Figure 1 illustrates the setup for the Small Size League. There is an overhead camera which captures video of the game and sends it to a computer. The computer then analyzes the images and sends out wireless signals to control the robots. The robots have unique colored markers to allow them to be visually identified. There are an extensive set of documents on the RoboCup website providing details on the rules of the game and constraints on robot design.

A central computer doing the processing for all of the robots offers a significant advantage over autonomous robots because it eliminates the need for communication and negotiation between independent agents. For example, if each robot was acting independently, in order to come up with an effective team strategy the robots

*Quoted from the RoboCup website at www.robocup.org

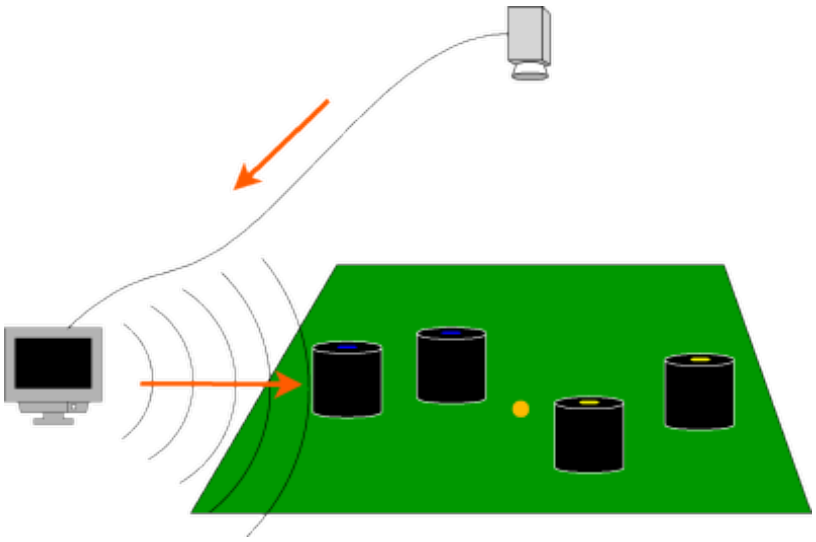


Figure 1. Arrows represent flow of information for Small Size League.

would need to communicate with each other and decide upon a single plan. The independent robots would each have a different view of the environment so they might have different opinions on what the best team strategy might be. Since the central computer has access to all of the information available it can determine the best strategy and issue commands to the robots. Besides the camera feed, another input that the central computer receives is data from a program called the RefereeBox. The RefereeBox is controlled by the human referee and communicates game information to the central computer. This game information contains data such as the current score and the current type of game play (such as penalty kick, kickoff, etc.).

1.3 Thunderbots Team

I am a member of UBC Thunderbots, a team of student volunteers at the University of British Columbia (UBC) which is aiming to compete in the Small Size League at RoboCup. We are divided into three smaller groups to focus on particular aspects of the competition. The mechanical engineering group is in charge of designing and building the physical robots. The image recognition (IR) team is writing software to analyze video from the camera to determine the location and orientation of the robots on the field. I am part of the AI team, which is writing software to control the robots by sending wireless signals based on the input from IR and the RefereeBox. Although these groups are working on independent tasks, there is still collaboration and communication between teams. Splitting into three groups also allows for better organization and clearly defines the goals that each team needs to work towards accomplishing.

1.4 Goals

Since the Thunderbots team plans to continue working on RoboCup soccer for several years, its important that the AI is designed in way that is easy to understand for new members and is also easily extensible for future growth and optimization. In order to be successful the AI also needs to satisfy certain basic requirements. One requirement is that it should be capable of processing a single time step at least as fast as the IR is capable of processing frames using a limited amount of computing resources. Ideally both the IR and the AI would be able to process frames at the same frame rate as the camera feed. The current Thunderbots camera captures video at 60 frames per second (FPS). Another requirement for the AI is that given an input from the IR and RefereeBox, it can send wireless signals to control the robots. Not only do these signals need to control the robots so that they do not violate any of the rules of the game, but they also need to make the robots effective soccer players capable of beating enemy teams.

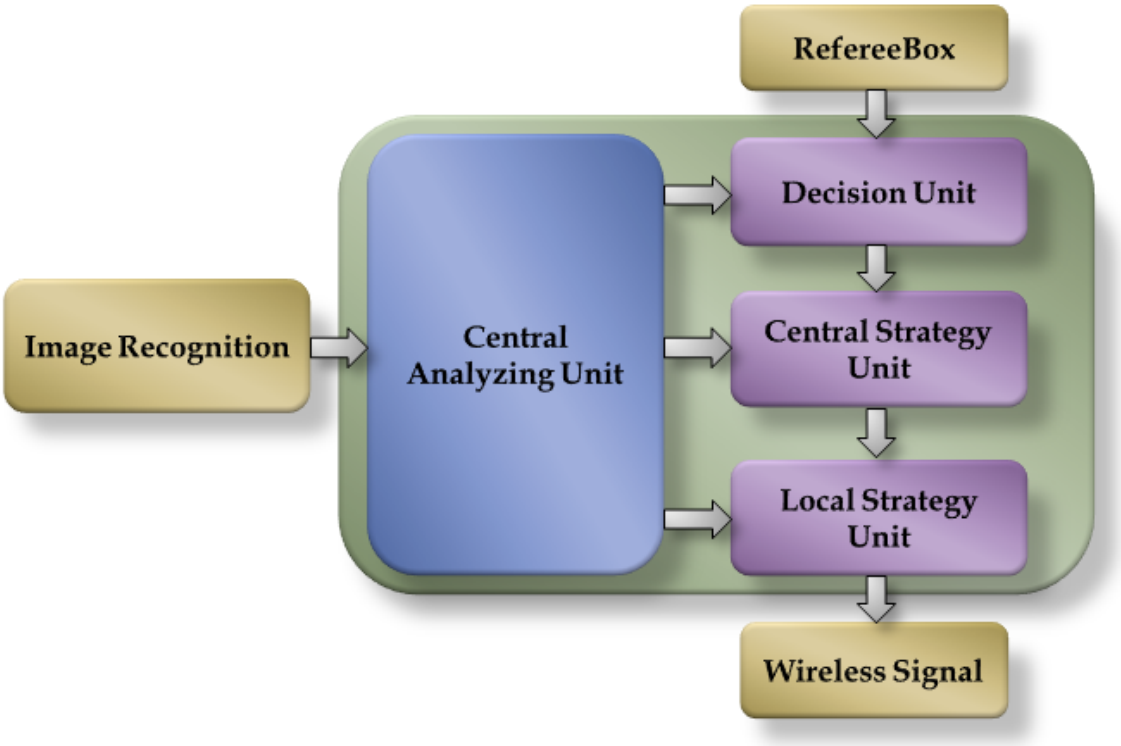


Figure 2. Hierarchical controller for Artificial Intelligence.

2. DESIGN PROCESS

Before beginning design or implementation of the AI, we first gathered information about the Small Size League in order to clearly understand the requirements of the system. Next, we compared the advantages and disadvantages of possible techniques we could use for creating the AI. We decided that instead of basing the system on a single AI technique, the problem could be split up into smaller subproblems. This would allow the application of various AI algorithms on the simpler subproblems. We also decided that we would start off by taking the lower risk approach of hardcoding the majority of the AI using human intuition. This approach would be more likely to result in a functional AI within the given time constraints than trying to apply AI techniques in an innovative way. Using principles from the field of software engineering such as the application of design patterns and object oriented programming, we then designed a high level system diagram. This system diagram represented the hierarchy responsible for controlling the robots.

2.1 Hierarchical Controller

Designing the AI with a hierarchy of different levels of abstraction offers many advantages over other methods of implementation. Each level of the hierarchy can be designed to focus on accomplishing a particular role with the assumption that the other layers that it interacts with accomplish their roles as well. For example, the layer in charge of controlling the global team strategy can assign lower level behaviors to robots without knowledge of how the behaviors are actually implemented. This hierarchy offers a clear advantage over other task oriented approaches by modularizing the different levels of abstraction. Implementing these independent modules is much simpler than tackling an entire task at once because each module represents a small part of the global problem. Another advantage of using a hierarchy is that from a software engineering perspective it makes the system more extensible and easier to understand. For example, instead of having to completely rewrite the code in order to create a new strategy, only a single layer at the top of the hierarchy would need to be modified.

Figure 2 illustrates the hierarchy used for our AI system. The AI has been split into five modules:

• **Central Analyzing Unit**: Receives and analyzes data from the IR module. This layer acts as an interface to the AI and controls the execution of the other layers.

• **Decision Unit**: Receives inputs from the RefereeBox and sets the global strategy type.

• **Central Strategy Unit**: Based on the global strategy type, a behavior is assigned to each robot.

• **Local Strategy Unit**: Based on the robot's behavior, lower level commands are sent to the Robot Controller.

• **Robot Controller**: Translates and transmits low level commands into wireless signals.

This design uses a hierarchy from a higher level strategy type (assigned in Decision Unit) to lower level robot behaviors (assigned in Central Strategy Unit) to the lowest level commands (assigned in Local Strategy Unit). The global strategy types include:

- Do nothing
- Get in starting positions
- Ball is in play
- Direct free kick
- Indirect free kick
- Penalty kick
- Throw in
- Goal kick
- Corner kick
- Victory dance

Behaviors assigned to robots include:

- Stop
- Pass ball to another player
- Receive pass from another player
- Shoot ball
- Chase ball
- Go to a location on the field
- Be the goalie

Low level commands include:

- Direction to move
- Direction to face
- Whether or not to dribble the ball
- Whether or not to kick the ball

There are hierarchies within the layers of the AI module as well. Within the Local Strategy Unit some behaviors are lower level than other behaviors. Higher level behaviors can be implemented using a combination of lower level behaviors. For example, to implement the shoot ball behavior, the chase ball behavior can be used if the robot does not have control of the ball. The chase ball behavior in turn uses the move behavior in order to get closer to the ball.

The various modules currently do not use any advanced AI techniques such as reinforcement learning or pattern recognition. However, these techniques could be applied within some of the modules. For example, machine learning techniques could be useful for assigning behaviors at the Central Strategy Unit level based

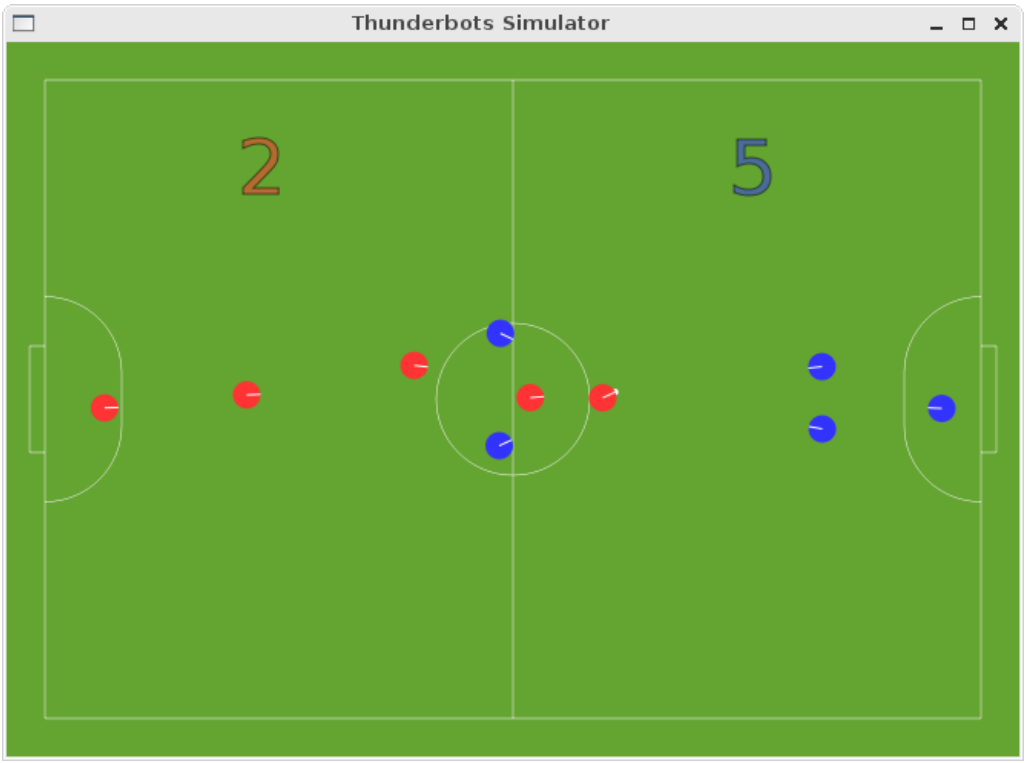


Figure 3. Screenshot of simulator.

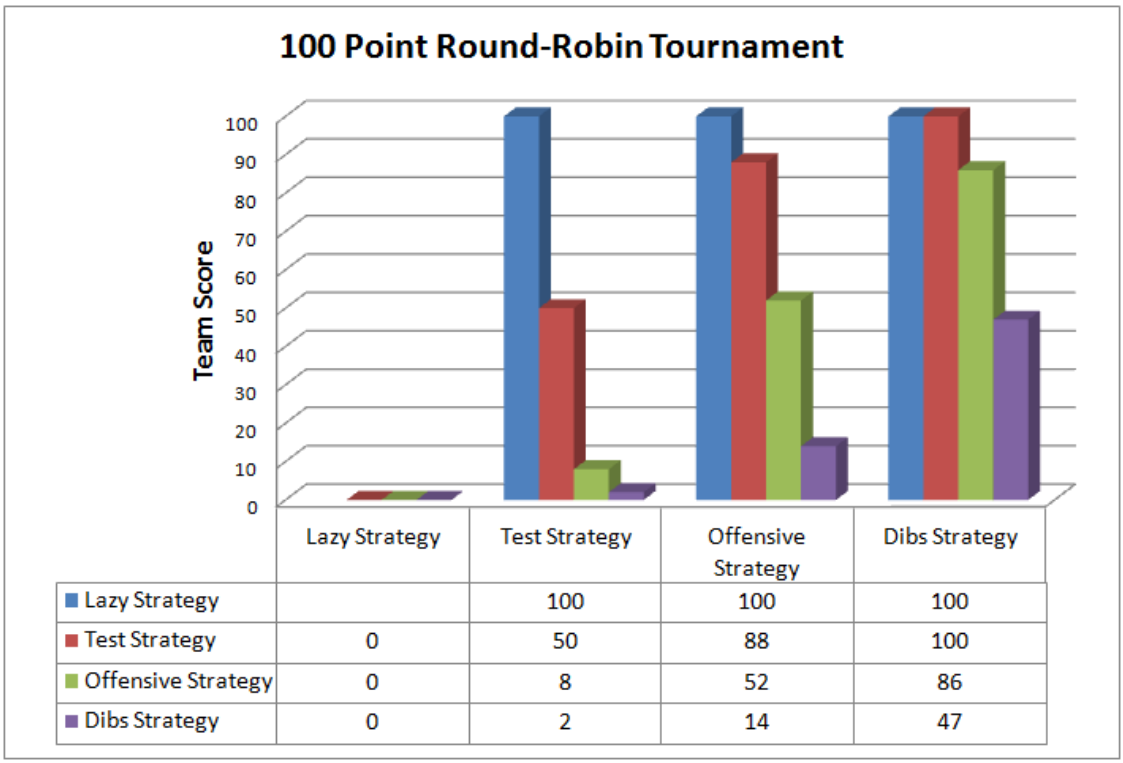
on the performance of the different strategies in simulations. One technique our current AI system uses is a simple physics engine which predicts where entities will be on the field at some time step in the future. These predictions can be extremely important for a variety of tasks in soccer such as obstacle avoidance, receiving passes, and blocking the goal. Originally we attempted to make these predictions using 'potential fields' which represented the probability that the ball would be at a particular location in the future. Further in the future these probabilities would become more spread out since it becomes harder to make predictions. After implementing and testing the potential fields we eventually decided to remove this feature since it turned out to be too computationally expensive. We now have a simpler point estimate of future locations which does not take into account uncertainty.

2.2 Simulator

The AI module also includes a simulator and a visualizer. The simulator is used to test robot strategies and behaviors without needing input from IR or outputting wireless signals to physical robots. It takes the place of all input and output to the AI (IR, RefereeBox, and wireless signals). In order to update the position of entities on the field it makes predictions based on velocity and acceleration. The Robot Controller updates the acceleration and properties of robots in the simulator instead of sending out wireless signals. The visualizer is a graphical interface which displays the position of the ball and robots on the field. The visualizer can be used for both simulations and actual games in order to observe the current state of the AI.

2.3 Strategies

Based on the global strategy type, the central analyzing unit can execute a particular type of strategy. These strategies are used to assign behaviors to the robots. For example, if the global strategy type is "get in starting positions" then each robot will be assigned a move behavior to the location of its starting position. For the global strategy type "ball is in play" there were a total of four strategies implemented:



	Lazy Strategy	Test Strategy	Offensive Strategy	Dibs Strategy
Lazy Strategy		100	100	100
Test Strategy	0	50	88	100
Offensive Strategy	0	8	52	86
Dibs Strategy	0	2	14	47

Figure 4. Results of a round-robin competition between four strategies in games consisting of 100 goals.

• **Lazy strategy**: One robot is assigned the goalie role and the other four robots always face the ball but do not move.

• **Test strategy**: One robot is assigned the goalie role and passes the ball to the closest player if it gets control of the ball. Two robots move back and forth from one side of the field to the other. The last two robots chase the ball and try to score a goal.

• **Offensive strategy**: One robot is assigned the goalie role and passes the ball to the closest player if it gets control of the ball. One defender robot moves back and forth between two points in parallel with its goal in an attempt to defend. If the defender gets the ball it passes to the closest player (excluding the goalie). The other three robots chase the ball and try to score a goal.

• **Dibs strategy**: Roles are assigned dynamically during the game. Players can switch roles depending on the state of the game. If a player is a good match for a particular role it claims 'dibs' on the role so that no other players can claim it. There are five roles: goalie, defender top, defender bottom, attacker, and supporter. The player closest to the ball is always the attacker and tries to score a goal. The player closest to the goal is always the goalie. The player closest to the top defender position is defender top, and likewise for defender bottom. The remaining player is the supporter and tries to assist the attacker in scoring a goal. The two defenders are more capable than the roaming defender used in the offensive strategy and are capable of coordinating positions based on the position of the ball relative to the goal. This strategy was designed based on a strategy described by Peter Stone [4].

3. RESULTS

The Thunderbots team did not qualify for the 2008 RoboCup open. The mechanical engineering team did not complete their design of the robots so we did not have any demonstration to show for qualification. The image recognition is also still in the early phases of development and is not yet capable of tracking robots in a real game environment at the required frame rate. The development of a simulator for the AI proved to be extremely useful

because it allowed development to continue even without input from a camera or physical robots to control. In the text based mode, our simulator is currently capable of running at 8000 FPS, well above the requirement of 60 FPS. After implementing the different levels of the AI framework we focussed on coding the lower level strategies such as movement. We invested the most amount of work into the movement function because it is used by almost every other low level behavior. It involves issues such as obstacle avoidance and path planning. It will probably need substantial modification after testing begins with physical robots since the conditions of the simulator probably do not match the conditions in a real world environment.

After successfully implementing the low level behaviors, the next task we concentrated on was the strategies in the Central Strategy Unit. These were fun to create because they are relatively easy to program since they only involve assigning behaviors to each robot. However, assigning the correct behavior can be a challenging problem which could determine the difference between a winning or a losing team. Sometimes the intuition behind which behavior to assign can be misleading and might not be the optimal solution. The simulator once again came in useful because it allows different strategies to be competed against each other. By the law of large numbers, after a long enough game a superior strategy will always beat an inferior one. Figure 4 shows the results of a round-robin competition performed between the different strategies we implemented. The dibs strategy was the clear winner.

One potential issue we considered about comparing strategies is the fact that it is theoretically possible that an inferior strategy can beat a superior one in certain situations. One reason this could occur is due to the stochastic nature of soccer games in a real world environment. Another reason that this could occur is that the inferior strategy could work well against only one specific type of strategy, and would not generalize well to the variety of strategies seen at the RoboCup competition. This fact might make it advantageous to create a collection of different strategies and then start dynamically changing strategies in the middle of a match based on the performance of the current strategy.

4. DISCUSSION AND FUTURE WORK

The results of the AI system so far seem promising. A better test of the effectiveness of the AI would be to test it in a real world environment with physical robots and IR. We have accomplished many of the goals we originally intended for the AI. It is capable of running at a very fast frame rate using limited computing resources. It also uses principles derived from software engineering to make the framework easy to learn for new users and also very flexible and extensible. This flexibility leaves lots of room for improvement and further optimization, especially for the use of more advanced AI techniques.

The AI is still not ready to be used in a competition. There are some rules which have not yet been implemented in the simulator. There are also some strategy types which have not been implemented in the Central Strategy Unit. For example, when the ball goes out of bounds it currently gets reset to the middle of the field and a kickoff takes place. These rules have not been implemented purely due to time constraints. There is nothing fundamentally difficult about implementing them. We decided to leave these tasks for later since they do not have a large impact on the game play. The Thunderbots team plans to continue development in an attempt to qualify for future competitions.

5. SUMMARY

The design process involved in creating the AI for RoboCup soccer has been a rewarding experience. Not only did it help reinforce skills related to software engineering, AI, and Computer Science in general but it also helped us learn about concepts used by the IR and mechanical engineering teams. The challenges involved in designing the system are not unique to the domain of robotic soccer and they can be easily generalized to more practical domains. For example, some of the challenges we encountered for navigation are universally common to robotics. The positive results obtained from our implementation of the hierarchical controller indicate that this framework can be successfully applied for other robotic systems as well.

ACKNOWLEDGMENTS

All of the members of the Thunderbots team contributed to making this project possible. Byron Knoll, George Stelle, and Kobe Lin designed and implemented the AI module. Alan Mackworth supervised Byron Knoll and provided guidance and support about different AI techniques that can be used for robotic soccer. The Thunderbots team was supported by UBC Thunderbird Robotics and their sponsors[†].

REFERENCES

[1] Barman, A. R., Kingdon, S. J., Mackworth, A. K., Pai, D. K., Sahota, M. K., Wilkinson, H., and Zhang, Y., "Dynamite: A testbed for multiple mobile robots," in [*Proc IJCAI Workshop on Dynamically Interacting Robots*], 38–45 (August 1993).

[2] Zhang, Y. and Mackworth, A. K., "A constraint-based robotic soccer team," *Constraints* **7**, 7–28 (2002).

[3] Lee, A., *Drill-Based Fitness Functions for Learning*, Honours thesis, The University Of British Columbia (April 2006).

[4] Stone, P., [*Intelligent Autonomous Robotics: A Robot Soccer Case Study*], Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers (2007).

[†]See www.ubcthunderbird.com