

Asynchronous Chess Competition

Nathaniel Gemelli, Robert Wright, James Lawton and Andrew Boes

US Air Force Research Lab
Rome Research Site
Rome, NY, USA

{Nathaniel.Gemelli, Robert.Wright, James.Lawton, Andrew.Boes}@rl.af.mil

ABSTRACT

Asynchronous Chess (AChess) is a platform for the development and evaluation of real-time adversarial agent technologies. It is a two-player game using the basic rules of chess with the modification that agents may move as many pieces as they want at any time. Modifying chess in this way creates a new robust, asynchronous, real-time game in which agents must carefully balance their time between reasoning and acting in order to out-perform their opponent. As a fast-paced adversarial game, many challenges relevant to real-world applications arise which give it merit for study and use.

Categories and Subject Descriptors

I.2.11 [Distributed Artificial Intelligence]: Intelligent Agents

General Terms

Experimentation, Algorithms

Keywords

Adversarial Environments, Real-Time Reasoning, Intelligent Agents

1. INTRODUCTION

We present Asynchronous Chess [2] as a real-time adversarial software-agent demonstration and competition for AAMAS 2006. Asynchronous Chess (AChess) is a platform for the development and evaluation of real-time adversarial agent technologies. It is a two-player (i.e. *agent*) game using the basic rules of chess with the modification that agents may move as many pieces as they want at any time. Modifying chess in this way creates a new robust, asynchronous, real-time game in which agents must carefully balance their time between reasoning and acting in order to out-perform their opponent. As a fast-paced adversarial

game, many challenges relevant to real-world applications arise which give it merit for study and use. We believe AChess will be of added value to researchers in the Artificial Intelligence field, and be well received as an AAMAS demonstration/competition.

2. ACHESS DESCRIPTION

AChess is a chess-like game. In the game, players are allowed to move as many pieces as they wish at any point during the game. It has the same board, the same pieces, and the same piece movement patterns as standard chess. The main difference between AChess and regular chess is that it removes the rule in which players must take turns. There are other differences between AChess and standard chess that were found to be necessary to make AChess a well-defined game. These differences include:

1. Any piece may be moved at any time as long as the piece is not already in the process of moving.
2. Pieces, including knights, traverse paths through squares to reach their destinations instead of teleporting.
3. Pieces traverse individual squares at specified rates that are specified based on the piece type (i.e. King, Queen, Bishop, Knight, Rook, Pawn).
4. Pieces cannot change directions while a move is being carried out. Agents must wait for a piece to finish its current movement before submitting a new move for the piece.
5. There is no check-mate. Kings must be captured.

3. ACHESS FEATURES AND CHALLENGES

AChess is a fully observable environment. A central server acts as a referee and supplies both players with a complete view of the game environment. This view includes the current positions of all the pieces and which pieces are currently in the process of moving. The only information agents do not have is where the moving enemy pieces are moving to and exactly how long it has been moving.

AChess is a stochastic environment. There is randomness in the AChess system. AChess simulates continuous time with extremely fine-grain discrete time steps. In the event that two or more pieces attempt to enter a square at the same time, the server randomly chooses one to enter the square. The other pieces are either killed or prevented from

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005 ...\$5.00.

moving. This challenges agents to handle the uncertainty in the environment.

ACheck is an episodic environment. Agents in competition will compete in a set of matches consisting of a series of games between two agents. This will give agents with learning capabilities a chance to repeatedly measure their performance against competitors and learn from their experience.

ACheck has an extremely large state and search space that the agents must manage. A normal game of chess has a search space size on the order of 10^{40} [1]. ACheck has an exponentially larger search space because of the ability to move more than one piece, and potentially all the pieces, asynchronously. In addition, pieces can be in a state of movement (*not even on a square!*) which drastically increases the size of the state space.

Finally, ACheck challenges agents with heterogeneous capabilities. The agents must effectively utilize pieces with different movement rates and movement patterns. This challenge is missing in many other competitive agent environments and is one of the more interesting features of ACheck.

4. ENVIRONMENT ARCHITECTURE AND LOGISTICAL REQUIREMENTS

The ACheck environment/server is implemented in Java and communicates to the agents with strings of text over standard TCP/IP sockets. This enables interested parties to develop their agent solutions using any language or platform they choose. It also allows the agents, environment, and visualizations to be resident on separate platforms, which in turn helps to prevent competitors from interfering with each other or the performance of the environment itself.

For the competition the matches will be run across four machines, two of which will be identical platforms for the agents (each team), over a local area network. One machine will host the ACheck server (the environment), and the last machine will host the visualizations. Four machines are required to enable each agent, the server, and the visualization to have their own computing platform. The machines will all be using the same operating system and be configured to be identical platforms for all competitors. The final decision on the operating system will be made at a later date and will be made based on competitor preference and logistic considerations.

5. COMPETITION DESCRIPTION

The competition will be run as a tournament consisting of rounds. Agent implementations will compete against one another in matches which will consist of a specified number of games. The final number of games in a match will be decided just before the start of the tournament and will be based on the number of entries. A large number of games per match are preferred, but the number of games may need to be small in the interest of time. The winner of a match will be determined by which agent was able to accumulate the most wins during the course of the match. Winners of matches will proceed to the next round until ultimately there is a single winner.

For the conference we will invite any and all interested parties to enter their own ACheck agent into the competition. Well before the conference a version of the ACheck environment will be provided to the public (via download) along with an API for agent development. Competitors will have to submit a final version of their agent to be used in the competition by a specified date.

In the competition the board will be the standard 8 by 8 square sized board. Pieces will be positioned in the standard chess layout at the start of the game. The piece rates, which determine how quickly pieces may traverse individual squares, will be defined at a later date.

6. REFERENCES

- [1] M. Beeler, R. W. Gosper, and R. Schroeppel. Hakmem. Technical report, Massachusetts Institute of Technology (TR AIM-239), Cambridge, MA, USA, 1972.
- [2] N. Gemelli, R. Wright, and R. Mailer. Asynchronous chess. In *Proceedings of The 2005 AAAI Fall Symposium Workshop on Coevolutionary and Coadaptive Systems*, November 2005.