Strategic Reasoning in Games

Caelan Alonge
Computer Science
Florida State University
ca10g@my.fsu.edu

Abstract:

As videogames become more and more complex, the variables required to obtain a win condition increase exponentially. In early games, the computer opponent may have had to use a simple predetermined path in hopes of a collision with the player. In today's real time strategy games, the computer opponent must use large sets of data containing information about resources available, map layout, player position, etc. to determine the best course of action. Strategic reasoning, or the agent's ability to use this information about the current game state to determine the next move, needs to be fast and efficient in order for the AI opponent to be a real challenge to the human player.

1. Introduction:

The strategies used by computer opponents in videogame scenarios are developed using the same basic methods regardless of the goal of the game itself. First the agent must gather all of the data from the game world required to determine the current game state. Then the agent must analyze the current game state and compare it with either a list of predetermined moves from that move, in the case of simple games such as tic tack toe, or, in the case of more advanced games such as today's real time strategy games, use complicated algorithms such as Monte-Carlo search or Alternating Time Temporal Logic to determine what set of moves brings them closer to a win condition on a large game tree.

Monte-Carlo search is a simulation based search paradigm that uses the outcomes of simulated games to determine the point values of completing certain actions in a real game.

Alternating-time temporal logic, (ATL) is a temporal logic that incorporates basic game theory. The computer agent's strategic ability is determined by answering certain questions about the current game state. These questions include:

How much of the game state is visible to the agent?

How much of the agents observations van be memorized?

2. Background:

Monte-Carlo search has been used in the past on games. This search method works better on smaller numbers of rollouts (simulated games) therefore it is ideal to use on complex games such as vast turn basted strategy games. Though many other algorithms have been used to compete with the built in AI of real time strategy games, Monte-Carlo search is the best tree search for large expansive games as it can be used to control all aspects of a game rather than certain specific aspects.

My task is to design a non-Linear Monte-Carlo search algorithm that will frequently win games against a computer opponent in the popular real time strategy game Starcraft II. While similar work has been done on turn based games such as Civilization II, my algorithm will compete against the computer player in a real time strategy game where the game is never paused and the game state is constantly changing.

3. Work:

Starcraft II is a real time strategy game. This differs from turn based strategy games in that there are no set turns. Each player does not get a set amount of time to determine the best strategy to defeat their opponent, rather all decisions must be made on the fly as both the human player and computer opponent make all their moves at the same time. The only real limiting factor into what moves a player can make at what time are the resources available to that player at that time.

Also unlike many turn based strategy games where the moves of one's opponents are clearly visible as they have their turn, real time strategy games like Starcraft II incorporate a "fog of war" feature which prevents one player from observing the moves of another unless they have units placed in line of sight, or they use some special observation ability. This means that a significant amount of strategic decision making comes from limited observations of the game state by scouting.

My prototype will use data from the Startcraft II game manual as the agent's knowledge database. The game state is to be defined by the attributes of the agent's current units, resources, and surroundings as well as information on the enemy's current units, resources, and surroundings given by a scout unit.

4. Experimental Setup:

I test my method on the latest version of Starcraft II Heart of the Swarm, with both players set as Terran for simplicity. The game is run on a desktop pc with Intel i7 CPU. The games last one hour. My method is compared against three baselines.

A human player: This person plays Starcraft II and knows the pros and cons of certain moves.

Another computer player: This computer player is hardcoded into the game by the developers and is identical to the opponent computer player.

A random command generator: This program randomly selects and issues commands from the list of commands currently available to the agent.

The performance of my algorithm is evaluated by the average win rate against Starcraft's AI computer opponent. Each method is run on three 1 hour games.

5. Results / Conclusions:

After running the trials with all methods, my method of non-Linear Monte-Carlo search defeated the computer opponent on average 65% of the time. The human player defeated the computer opponent 70% of the time, the built in AI 45% and the random command generator 9%.

As the non-linear method was almost as effective as a human player and 20% more effective than the games built in AI, non-Linear Monte-Carlo search proves to be a very effective search function for complex fast paced games.

6. References

[Johanson, 2011] Michael Johanson "Accelerating Best Response Calculation in Large Extensive Games" http://ijcai.org/papers11/Papers/IJCAI11-054.pdf

[Jamroga, 2011] Wojciech Jamroga "Comparing Variants of Strategic Ability" http://ijcai.org/papers11/Papers/IJCAI11-053.pdf

[Michael 2011] Loizos Michael "Causal Learnability" http://ijcai.org/papers11/Papers/IJCAI11-174.pdf

[Branavan 2011] S.R.K. Branavan "Non-Linear Monte-Carlo Search in Civilization II" http://ijcai.org/papers11/Papers/IJCAI11-401.pdf

[Gibson 2011] Richard Gibson "Regret Minimization in Multiplayer Extensive Games" http://ijcai.org/papers11/Papers/IJCAI11-473.pdf