AI Model for Computer games based on Case Based Reasoning and AI Planning

Vlado Menkovski
Athens Information Technology
0.8km Markopoulou Ave.
Peania, 19002, Greece
vmen@ait.edu.gr

Dimitrios Metafas
Athens Information Technology
0.8km Markopoulou Ave.
Peania, 19002, Greece
dmeta@ait.edu.gr

Abstract

Making efficient AI models for games with imperfect information can be a particular challenge. Considering the large number of possible moves and the incorporated uncertainties building game trees for these games becomes very difficult due to the exponential growth of the number of nodes at each level. This effort is focused on presenting a method of combined Case Based Reasoning (CBR) with AI Planning which drastically reduces the size of game trees. Instead of looking at all possible combinations we can focus only on the moves that lead us to specific strategies in effect discarding meaningless moves. These strategies are selected by finding similarities to cases in the CBR database. The strategies are formed by a set of desired goals. The AI planning is responsible for creating a plan to reach these goals. The plan is basically a set of moves that brings the player to this goal. By following these steps and not regarding the vast number of other possible moves the model develops Game Trees which grows slower so they can be built with more feature moves restricted by the same amount of memory.

Categories and Subject Descriptors

I.2.1 [Applications and Expert Systems]: *Games*

General Terms
Algorithms, Performance.

Keywords
Game AI, Case Based Reasoning, AI Planning, Game Trees

1. Introduction

The goal of this effort is to explore a model for design and implementation of an AI agent for turn based games. This model provides for building more capable computer opponents that rely on strategies that closely resemble human approach in solving problems opposed to classical computational centric heuristics in game AI. In this manner the computational resources can be focused on more sensible strategies for the game play.

With the advancement in computer hardware increasingly more computing power is left for executing AI algorithms in games. In the past AI in games was mainly a cheating set of instructions that simulated the increasing difficulty in the game environment so that the player had the illusion of real counterpart. Improvement in available memory and processing power allows implementation of more intelligent algorithms for building the game environment as well as direct interaction with the human players.

In this particular research the emphasis is put on the interaction between the AI agent and a computer player in the realm of the game rules. It is particularly focused on turn based games that have the elements of uncertainty like dice or concealed information. At the beginning a description of Game AI algorithms are given; such as Game Trees and Minimax. The following section describes an approach of using AI Planning to improve building Game Trees in games with imperfect information where Game Trees tend to be very large with high growth ratio. Section 4 discusses another approach that provides a significant reduction to the number of considered moves in order to find the favorable strategy of the AI player. This approach uses AI Planning techniques and Case Base Reasoning (CBR) to plan for different scenarios in predetermined strategies which would be analogous to human player experience in the particular game. The CBR database illustrates a set of past experiences for the AI problem and the AI Planning illustrates the procedure to deal with the given situation in the game. In the next two sections implementations and evaluations of both approaches are given. The AI Planning approach is implemented with the Tic-tac-toe game and the combined AI Planning and CBR approach is implemented with a model for the Monopoly game. The last part contains conclusions and future work ideas.

2. Game Trees and Minimax

Game Trees are common model for evaluating how different combinations of moves from the player and his opponents will affect the future position of the player and eventually the end result of the game. An algorithm that decides on the next move by evaluating the results from the built Game Tree is minimax [1]. Minimax assumes that the player at hand will always choose the best possible move for him, in other words the player will try to select the move that maximizes the result of the evaluation function over the game state. So basically the player at hand needs to choose the best move overall while taking into account that the next player(s) will try to do the same thing. Minimax tries to maximize the minimum gain. Minimax can be applied to multiple