**Moves Performance Optimisation for Large Search Space using Decision Tree Learning Technique on Minimax Algorithm with Alpha-Beta Pruning**

**Khong Wai Foo Andrew,** **Sun Yick William, Samuel Winterton, Kim Tae Hyung**School of Computer Science & Software Engineering  
The University of Western Australia  
35 Stirling Highway, Crawley WA 6009  
[khongw01@student.uwa.edu.au](mailto:khongw01@student.uwa.edu.au1), [suny05@student.uwa.edu.au](mailto:suny05@student.uwa.edu.au2), [wintes02@student.uwa.edu.au](mailto:wintes02@student.uwa.edu.au3), [kimt01@student.uwa.edu.au](mailto:kimt01@student.uwa.edu.au4)

**Abstract – The Minimax algorithm with alpha-beta pruning (or simply alpha-beta pruning) is commonly used in machine playing of two-player games such as Tic-tac-toe and chess.**(1)

**In this paper, we show that an agent using the alpha-beta pruning algorithm will not win the game or wins the game with longer time duration than it should take with a large moves space in the MixMeta4 environment. A decision tree learning technique will then be used to decide the move that will lean towards winning the game in the fastest possible way. Finally data gathered from the decision tree will be compared with just the alpha-beta pruning algorithm to determine whether there is an improvement in the agent’s play.**

**[NEED TO INCLUDE WHAT WE FOUND HERE]**

*Keywords:* Minimax, Alpha-Beta Pruning, MixMeta4, Decision Tree, Learning

**1 Introduction**

The Minimax algorithm with alpha-beta pruning or simply just alpha-beta pruning algorithm is more commonly used in machine playing games than the naïve Minimax algorithm, generally performing better by pruning away search paths and thus reducing the size of the search space.

In the MixMeta4 environment, an agent that utilises the alpha-beta pruning algorithm is expected to win a game against an agent that simply chooses random moves. However, when played against more intelligent agents such as Hal, its endgame performance is lacking, often producing moves backwards or away from the opposition, losing the game.

In this paper, we investigate the effect of allowing the agent to learn better moves from playing agents such as Hal, where the result probabilities are gathered using Decision Trees and stored for future games.

Therefore our hypothesis is, that an Agent that uses the Decision Tree learning technique will make beneficial moves that will improve its game playing performance over an agent that simply uses only Alpha-Beta Pruning.

In section 2 we describe our method for capturing *good* moves using Decision Tree learning, and in section 3 show the effect of applying this *learnt knowledge* against agents such as Hal that have historically performed well against a Alpha-Beta Pruning agent.

Finally we discuss in a more general sense what implications our results have for not only game playing, but also machine learning.

**2 Method**

**2.1 Learning Decision Trees**

To effectively "teach" our Agent what the best move is given the current state of the environment, we asked it to "learn" decision trees, something known as Decision Tree Induction.

Each node within the tree is called an Attribute, and can be thought of as the input. Each resulting decision is called the Goal.

We started by building a training set; a database of Attributes and their associated Goals.

To maintain a feasible experiment we chose relatively simple, but pertinent Attributes to compute, such as if our Agent could take a piece or be taken by a piece.

The Goals were provided by a human "expert" or an Agent known to be better than Alpha-Beta pruning. Originally we hoped to provide all Goals via human experts, but due to the volume of data, generated Goals from known better performing players were used.

Once the training set was complete for multiple games, we then analysed the sets together to produce probabilities for each Decision Tree branch. This data was then fed into our utility function within our Alpha-Beta pruning algorithm.

[NEED MORE HERE ABOUT LIBRARY ETC] **2.3 Application of Moves**

**3 Results**

**4 Discussion  
4.1 Alpha-Beta Pruning  
4.2 Game Playing  
4.3 Decision Trees  
4.4 Machine Learning  
  
5 Conclusion**

**References**

1. Alpha-Beta pruning. *Wikipedia.* [Online] [Cited: May 13, 2009.] http://en.wikipedia.org/wiki/Alpha-beta\_pruning.