Adapting MP-Mix Search and Using Reinforcement Learning for Drafting Games

CMPUT 651 Project Proposal

Neesha Dessai, Richard Gibson and **Richard Zhao**
Department of Computing Science, University of Alberta
Edmonton, Alberta, T6G 2E8, Canada
{neesha|rggibson|rxzhao}@cs.ualberta.ca

Introduction

Many commercial video games on the market today, particularly sports games, include some kind of drafting aspect. A draft works by having multiple agents take turns selecting from a collection of items until either agent has made a fixed number of choices or there are no more items left. For instance, when playing through multiple seasons of the baseball game *MLB 09: The Show* from Sony, a rookie draft takes place in between each season, where the human player and the computer opponents take turns choosing from a pool of new players who may then be added to the respective team's roster. Another example is the new hockey game *NHL 10* by EA Sports, where a fantasy draft can occur at the beginning of a season. In a fantasy draft, all players are removed from their current teams, and then selected one-by-one by the agents (human or computer-controlled teams) in the league.

What is the best strategy for making selections in a draft in order to have the "best" set-up (or team of players) we could possibly achieve? We expect to develop a method, perhaps search-based or learning-based, which will result in "better" drafting than current methods used in our domain of choice, Risk.

Problem Formulation

We will use the board game *Risk* by Hasbro Inc. as a testbed for our drafting technique. In the game of Risk, there are two main variations that can be used to start the game. The first variation randomly assigns each of the 42 territories among the (up to six) players so that each player has an equal (or as close as possible) number of territories. Another, more strategically demanding opening is to use territory selection, where the players particapte in a draft until all territories have been selected. The latter case is what we will use throughout this project. Note that we will be using uniform initial army placements, as is done in (Zuckerman, Felner, & Kraus 2009).

Given that we will be experimenting with drafting strategies in Risk, how can we determine if our draft selections were "good" or "bad"? In idealistic terms, we can measure this using an oracle function which returns our probability of winning the game, given the outcome of the draft and the strategies used following the draft. Such an oracle function is not feasible to calculate, as the game of Risk is extremely large and assumptions must be made about our opponents' strategies; however, game state evaulation functions are commonly computed to estimate positions in games. For instance, if v_i is the value to player i of the initial state following the draft, then player j could approximate the oracle function via

$$\frac{v_j}{\sum_i v_i}.$$

Our problem is to determine how we should select territories in the draft as to maximize this final oracle value, or perhaps some approximation of it.

Related Work

There has been some work towards creating competent bots for playing the game of Risk. Firstly, (Johansson & Olsson 2006) (and in more detail in (Olsson 2005)) use a handcrafted territory value function to determine the desirablity of conquering each of the enemy territories. This was used to determine whether or not to continue to attack, and if so, which of the territories to attack. Then, (Zuckerman, Felner, & Kraus 2009) developed a search strategy called MP-Mix which utilizes the evaulation function of (Johansson & Olsson 2006) to perform a type of extension of Minimax search to multi-player games. While these previous efforts have shown good results in Risk with randomly assigned initial territories, no work has been done to our knowledge towards good techniques for drafting in Risk. In addition, we are not aware of any artificial intelligence research directly related to drafting in another domain. Our problem appears to be fairly unexplored.

As we begin to experiment with our ideas as described in the next section, we will likely want to find related research which uses these ideas, even if not directly applied to drafting. Thus our library of related work will surely expand as the project develops.

Proposed Approach

We have two main ideas on how to approach drafting in Risk:

1. Adapt MP-Mix search (Zuckerman, Felner, & Kraus 2009) to drafting. One issue with MP-Mix in post-draft Risk is that the branching factor is restricted to the 3 most

promising moves so that the size of the search tree is controlled. However, with drafting, there will often be more than 3 moves that look very promising, in particular at the beginning of the draft when all territories are available. Once adapting MP-Mix search to simply apply to the drafting game, can we dynamically control the branching factor of the search? Also, can we use the approximate oracle function idea to evaluate our search at the terminal nodes in the draft? Furthermore, could we abstract the draft states to simplify MP-Mix search in a similar manner to PRA*? A natural way of grouping territories together in Risk is by the continent which they belong to, as there are bonuses for owning each of the entire continents. Could this type of abstraction simplify the search?

2. Use reinforcement learning and repeated self-play to develop a solid drafting strategy. Again, this would require some sort of draft state abstraction, as there are simply too many ways a draft can go down. What type of learning algorithm would be best suited for drafting? What should be used as the reward signal? Again, we could try to use an approximation of an oracle function, or we could "roll out" the rest of the game following some base strategy and use the win/loss result as the reward. This roll out method may be more desirable as it would give us a draft strategy that works well with the strategy played post draft, something that an approximation of an oracle function may not be able to do.

We hope to show that our final drafting strategy works better than the hard-coded strategies currently used for AI bots in Risk.

Theoretical Analysis

When playing Risk against a computer controlled opponent, we do not want to be forced to wait long periods of time for the computer to decide on an action. We must therefore take the runtime into account for our approach. If we decide to use reinforcement learning, this should not be an issue, as RL techniques are generally designed to allow for immediate action response to a given observation. On the other hand, with search, we may need to include a time limit before requiring an action to be selected. Unfortunately, (Zuckerman, Felner, & Kraus 2009) do not provide any details about such time limits.

At the moment, we cannot think of any other theoretical properties we wish to explore with this project.

Empirical Evaluation

We will use the Lux Delux[1] environment to run all of our experiments. Lux Delux provides several Risk bots with source code, as well as an API for writing our own agents. To evaluate our drafting strategies, we will compare them to the drafting strategies used by bots within Lux by playing matches. To do so, we will keep the post-draft strategy constant for all players, so that the drafting stage is the only difference in strategies among the players. Initially, we will use the post-draft strategies provided with Lux, and perhaps later try out our own. Note, however, that it is important that the drafting strategy plays well with the post-draft strategy. For instance, if our post-draft strategy is to conquer Europe but we claim South America in the draft, then this overall strategy is likely to perform very poorly (compared to a post-draft strategy which tries to keep South America). We plan to report our results in much the same manner as in (Zuckerman, Felner, & Kraus 2009).

[1]http://sillysoft.net/lux/

Future Work

While we strive to find an approach that works well for drafting in the Risk domain, our final product may not extend well to larger domains, such as fantasy drafts in sports video games. This is because fantasy drafts typically involve many more selections than drafts in Risk, and the pool of players is much greater than 42, the number of territories in Risk. So we could see possible future work adapting our project to work for a larger variety of domain sizes. However, we hope that our techniques we develop for drafting will still scale well to large environments.

References

Johansson, S., and Olsson, F. 2006. Using multi-agent system technology in risk bots. *In AIIDE*.

Olsson, F. 2005. A multi-agent system for playing the board game Risk. Master's thesis, Blekinge Institute of Technology.

Zuckerman, I.; Felner, A.; and Kraus, S. 2009. Mixing search strategies for multi-player games. *In IJCAI*.