Wizard Card Game

CS39440 Major Project Report

Author: Eithen Howard (esh3@aber.ac.uk)
Supervisor: Dr/Prof. Christine Zargesr (chz8@aber.ac.uk)

21th February 2019
Version: 1.0 (Draft)

This report was submitted as partial fulfilment of a BSc degree in Computer Science (G400)

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB
Wales, U.K.

Declaration of originality

I confirm that:

- This submission is my own work, except where clearly indicated.
- I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
- I have read the regulations on Unacceptable Academic Practice from the University's Academic Registry (AR) and the relevant sections of the current Student Handbook of the Department of Computer Science.
- In submitting this work I understand and agree to abide by the University's regulations governing these issues.

Name Eithen Dennis Steven Howard

Date .....

Consent to share this work

By including my name below, I hereby agree to this project's report and technical work being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name Eithen Dennis Steven Howard

Date .....

Acknowledgements

I am grateful to...

I'd like to thank...

Abstract

For this project I will be creating a Wizard Card Game which will be used to have an environment for a Monte Carlo Tree Search Artificial Intelligence. Wizard is a Card Game which a normal 52 card deck and 4 wizard and jester cards. Wizards are the highest and jesters are the lowest cards in the game. In the game bid for a round and try to win as many tricks in that round as possible, if you′re closer to your original bid the more point you will score. A trick can be won by having the highest card that match the trump suit (the card played after the hands are handed out) or the dealer′s suit. Another way to win is by play a wizard card which trumps all other cards. Once the rounds are over the person with the highest score wins.

The Monte Carlo Tree Search is the AI technique I will be using which involves creating a tree from the state that the game is currently in. This is then randomly expanded by selecting a card to play. It will then select a state from the ones that have been expanded. This will then simulate a game and will provide a score for how close it was to the original bid and give the score to the original child state. This is then repeated for a specified amount of time, then the one that has the highest ratio of score to how many times it is visited, this will then give the best card to be played.

Contents

List of Figures

List of Tables

Chapter 1

Background & Objectives

1.1 Background

In this section I will be talking about the Wizard Card Game, how the game works and how you can win the game and show in the this video [1]. I will also discuss the Artificial Intelligence techniques that were consider using for the opponents, which are neural networks, minmax and Monte Carlo algorithms. This will be expanded upon to how I concluded using the Monte Carlo Tree search for this game.

1.1.1 Wizard Card Game

1.1.1.1 Dealing and Setup

Wizard is a card game that contains 52 normal deck cards and 8 extra cards 4 of which are called wizards and the other 4 are called jesters [2]. Each player is dealt the same number of cards that is the round. For example, round 1 means each person get 1 card but round 10 means they each get ten cards. This is then played until the is no cards left to be dealt out mean for 3 people there should be 20 round and 15 rounds for 4 people. A trump card is also dealt from the deck at the beginning of the round from the top. After each play in a round the dealer is changed clockwise.

1.1.1.2 Playing and Bidding

One the trump has been dealt, the person to the left of the dealer states how many tricks they think they will win in this round. The maximum they can bid it the number of cards they have in their hands. Once everyone has put their individual bids in, the player to the left of the dealer puts down the first card which can be any card they want, then each player can either match the suit that the first player put down or the better play it to match the trump suit or play a wizard to win. Otherwise they can play off suit or play a jester to lose.

1.1.1.3 Scoring

To score points in this game you must use the bid on how many tricks you think you can win, giving you 20 point for being right and 10 point for every trick you win. Otherwise you get negative 10 point per every trick you were over or under for that round. For example, if you guess 4 tricks to win and only one 1 you would lose 30 points.

1.1.2 Monte Carlo Alogrithm

The Monte Carlo Algorithm is probilistic search alogrithm that creates a tree based on the current states of the problem. This is then randomly expanded from the root node and simulated what might happen, under specfic constrains such a iterations of the simulation or under an amount time. This will then gvie a score back on what how well the simulation wen. once this is done the best option will be picked based on what has the higest score from the tree that has currently been created. The next sections belows wil show the 4 phases that are used in the algorithm.

1.1.2.1 Selection

For this plase in the algorithms it will start with the root node that shows the intial state of the game. This will then select the child node that has the best win rating.

1.1.2.2 Expansion

1.1.2.3 Simulation

1.1.2.4 Backpropagation

1.1.3 Similar Algorithms

In this section, I will be talking abolu the two other Algorithms that could of been used for the Artifical intelligence of the game. I will also discuss why they werent used and why the Monte Carlo Tree Search was the one i picked.

1.1.3.1 Neural Network

Neural Network are artifically created based on the neuronal structure of the mamalian cerebral cortex but on much smaller scales. [3]

1.1.3.2 MinMax

1.2 Analysis

In this section I will be talking about what I will be aiming doing achieve with this product and what technology I will be using during this process.

1.2.1 Project Aims

The aim for this project is create a functional software version of the card game Wizard. This will also contain an Artifical intelligence Opponent that uses the Monte Carlo Algorithm to select a card for that round. The game will be simplified to only have 3 players, two of whihc will be the AI and one which is the human player. Each player will have 15 cards each and play until there are no cards left for 5 rounds.

1.2.2 Technology Overview

1.2.2.1 Programming Language

Use Java

1.2.2.2 Hardware

Say Specifiction of Harware used, and not need be helpful for faster processing speed.

1.3 Research Method and Software Process

1.3.1 Foreseen Challenges

applying Rules, Implementiong MonteCarlo Tree search and optimising

1.3.2 Process Methodology

There was a lot of options for developing my project. In the end I decide to you a agile approach based on Scrum. This would give me an iterative way of progressing in the project, use the sprints in the methodology to progressively improve on the previous work that I have done, adding more functionality and complexity to it as it is being developed.I feel this was a better option that a traditional process such as the waterfall model. This is because I can start work on the software whilst writing up the report without having to do redo all the work I have previously done when changes are made to the design of the software.

1.3.2.1 Version Control

Chapter 2

Design

You should concentrate on the more important aspects of the design. It is essential that an overview is presented before going into detail. As well as describing the design adopted it must also explain what other designs were considered and why they were rejected.

The design should describe what you expected to do, and might also explain areas that you had to revise after some investigation.

Typically, for an object-oriented design, the discussion will focus on the choice of objects and classes and the allocation of methods to classes. The use made of reusable components should be described and their source referenced. Particularly important decisions concerning data structures usually affect the architecture of a system and so should be described here.

How much material you include on detailed design and implementation will depend very much on the nature of the project. It should not be padded out. Think about the significant aspects of your system. For example, describe the design of the user interface if it is a critical aspect of your system, or provide detail about methods and data structures that are not trivial. Do not spend time on long lists of trivial items and repetitive descriptions. If in doubt about what is appropriate, speak to your supervisor.

You should also identify any support tools that you used. You should discuss your choice of implementation tools - programming language, compilers, database management system, program development environment, etc.

Some example sub-sections may be as follows, but the specific sections are for you to define.

2.1 Overall Architecture

2.1.1 Class Descriptions

2.1.1.1 Game Classes

Card

Deck

Player

Round

Rules

Game

Suit

Main

2.1.1.2 Aritifical Intelligence Classes

Node

Tree

GameState

MonteCarloTreeSearch

UCT

2.1.2 Data Flow Diagrams

2.2 Some detailed design

In this section i will outline the important Algorithms to be used in the program and discuss how the User interface will looked and function.

2.2.1 Algorithm Design

2.3 User Interface

2.4 Other relevant sections

Chapter 3

Implementation

3.1 Introduction

3.2 Sprints

Chapter 4

Testing

Detailed descriptions of every test case are definitely not what is required here. What is important is to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on “real users”? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

Whilst testing with “real users” can be useful, don’t see it as a way to shortcut detailed testing of your own. Think about issues discussed in the lectures about until testing, integration testing, etc. User testing without sensible testing of your own is not a useful activity.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

4.1 Overall Approach to Testing

4.2 Automated Testing

4.2.1 Unit Tests

4.2.2 User Interface Testing

4.2.3 Stress Testing

4.2.4 Other types of testing

4.3 Integration Testing

4.4 User Testing

Chapter 5

Evaluation

5.1 Introduction

5.2 Aim

5.3 Objectives

5.4 Design

5.5 Project Management

5.6 Testing

5.7 Further Development

Examiners expect to find a section addressing questions such as:

- Were the requirements correctly identified?
- Were the design decisions correct?
- Could a more suitable set of tools have been chosen?
- How well did the software meet the needs of those who were expecting to use it?
- How well were any other project aims achieved?
- If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

The questions are an indication of issues you should consider. They are not intended as a specification of a list of sections.

The evaluation is regarded as an important part of the project report; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things in the work and aspects of the work that could be improved. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

Annotated Bibliography

[1] "How to play:wizard video," July 2014. [Online]. Available: www.youtube.com/watch?v=kgRoZ-Yd7tg

[2] "Wizard card game desciptive overview," accessed in March 2019. [Online]. Available: https://boardgamegeek.com/boardgame/1465/wizard

[3] "Introduction to neural networks," 2019, accessed: 11th April. [Online]. Available: http://pages.cs.wisc.edu/~bolo/shipyard/neural/local.html

[4] "Shuffling algorithm," accessed in March 2019. [Online]. Available: www.geeksforgeeks.org/shuffle-a-given-array-using-fisher-yates-shuffle-algorithm/

[5] "Mcts description," accessed in March 2019. [Online]. Available: mcts.ai/about/index.html

Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

For example, if you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document as an appendix. In the body of your report you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.

If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification. Perhaps you use stories to keep track of the functionality and the 'future conversations'. It might not be relevant to include all of those in the body of your report. Instead, you might include those in an appendix.

There is a balance to be struck between what is relevant to include in the body of your report and whether additional supporting evidence is appropriate in the appendices. Speak to your supervisor or the module coordinator if you have questions about this.

Appendix A

Third-Party Code and Libraries

If you have made use of any third party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. If third party code or libraries are used, your work will build on that to produce notable new work. The key requirement is that we understand what your original work is and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

The following is an example of what you might say.

Apache POI library - The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the client's existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [**?**]. The library is released using the Apache License [**?**]. This library was used without modification.

Include as many declarations as appropriate for your work. The specific wording is less important than the fact that you are declaring the relevant work.

Appendix B

Ethics Submission

This appendix includes a copy of the ethics submission for the project. After you have completed your Ethics submission, you will receive a PDF with a summary of the comments. That document should be embedded in this report, either as images, an embedded PDF or as copied text. The content should also include the Ethics Application Number that you receive.

Appendix C

Code Examples

For some projects, it might be relevant to include some code extracts in an appendix. You are not expected to put all of your code here - the correct place for all of your code is in the technical submission that is made in addition to the Project Report. However, if there are some notable aspects of the code that you discuss, including that in an appendix might be useful to make it easier for your readers to access.

As a general guide, if you are discussing short extracts of code then you are advised to include such code in the body of the report. If there is a longer extract that is relevant, then you might include it as shown in the following section.

Only include code in the appendix if that code is discussed and referred to in the body of the report.

3.1 Random Number Generator

The Bayes Durham Shuffle ensures that the psuedo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs [**?**].

```
#define IM1 2147483563
#define IM2 2147483399
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
#define IA1 40014
#define IA2 40692
#define IQ1 53668
#define IQ2 52774
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
```

```
#define RNMX (1.0 - EPS)

double ran2(long *idum)
{
  /*-------------------------------------------------*/
  /* Minimum Standard Random Number Generator        */
  /* Taken from Numerical recipies in C              */
  /* Based on Park and Miller with Bays Durham Shuffle */
  /* Coupled Schrage methods for extra periodicity   */
  /* Always call with negative number to initialise  */
  /*-------------------------------------------------*/

  int j;
  long k;
  static long idum2=123456789;
  static long iy=0;
  static long iv[NTAB];
  double temp;

  if (*idum <=0)
  {
    if (-(*idum) < 1)
    {
      *idum = 1;
    }else
    {
      *idum = -(*idum);
    }
    idum2=(*idum);
    for (j=NTAB+7;j>=0;j--)
    {
      k = (*idum)/IQ1;
      *idum = IA1 *(*idum-k*IQ1) - IR1*k;
      if (*idum < 0)
      {
        *idum += IM1;
      }
      if (j < NTAB)
      {
        iv[j] = *idum;
      }
    }
    iy = iv[0];
  }
  k = (*idum)/IQ1;
  *idum = IA1*(*idum-k*IQ1) - IR1*k;
  if (*idum < 0)
  {
```

```
    *idum += IM1;
  }
  k = (idum2)/IQ2;
  idum2 = IA2*(idum2-k*IQ2) - IR2*k;
  if (idum2 < 0)
  {
    idum2 += IM2;
  }
  j = iy/NDIV;
  iy=iv[j] - idum2;
  iv[j] = *idum;
  if (iy < 1)
  {
    iy += IMM1;
  }
  if ((temp=AM*iy) > RNMX)
  {
    return RNMX;
  }else
  {
    return temp;
  }
}
```