Technical University of Crete

School of Electrical and Computer Engineering

Design and Implementation of an Automated Bridge Playing Agent

Marios Asiminakis

Examination Committee:

Georgios Chalkiadakis - Associate Professor (Supervisor)
Michail Lagoudakis - Associate Professor
Antonios Deligiannakis - Associate Professor

Chania - January 2017

Abstract

The game of bridge is one of the most challenging card games, making it a mental sport like chess. Building an automated bridge player is a great challenge in the field of artificial intelligence because the players lack perfect information. A game of bridge consists of the bidding phase and the playing phase. In this thesis we mainly addressed the playing phase. We implemented an agent that solves the perfect information variant of bridge, called double-dummy bridge. In order to achieve that, we had to theoretically prove the correctness of our methods. Afterwards, we designed and implemented methods for overcoming the problem of imperfect information from the declarer's point of view. Three different agents were implemented, one based on the widely used Monte-Carlo approach, while the other two are based on a novel method which gathers information more effectively. We also implemented two agents that played as defenders. While they both play with perfect information, one of those tries to exploit the lack of information for the declarer. We studied the effectiveness of our agents by testing them in a wide range of game scenarios, with very encouraging results.

Contents

Part I

INTRODUCTION, BACKGROUND AND THE RULES OF BRIDGE

Chapter 1

INTRODUCTION

The goal of this thesis is the design and the implementation of an agent that plays bridge, a well known card game. Bridge is considered one of the hardest and most technical card games. Bridge is a game that is played by two partnerships of two players (also called pairs), one being North-South and the other beeing East-West, named after the cardinal direction of their seat in the table. The goal of each pair is to maximize their score (which is the opossite of their opponents' score). The score is detrmined by the contract[1] and the number of tricks[2] won by each pair. Imperfect information and the existence of partnerships (as oposed to players) make it differ from most games.

Bridge is played in two phases. The first is the bidding phase and the second one the playing phase. In the bidding phase each player tries to communicate with their partner in order to reach a good contract and interfere with the opponents comunication in order to prevent them from reaching a good contract. Communication with the partner is only allowed through specific calls (bids, double, redouble, pass) and most calls are legal only under certain circumstances. A call by one player might make some calls illegal for the opponents, limiting their options and disrupting their communication. Each partnership use some conventions concerning the bidding. Those are agreements concerning the meaning of each call. For example, a common agreement states that a bid of 2♣ when all the previous players have passed promise a very strong hand. The outcome of the bidding is the contract which determines the declarer, the trump suit and the association between the number of tricks won to the final score.

In the playing phase each player tries to maximize the number of tricks for his partnership, based on the decided contract. Different contracts, even with the same cards, might lead into different lines of play. This is because a different trump suit can radically change the game and the different trick to score mappings can cause a more passive or a more agressive line of play. The two phases of bridge are distinct but not independent. In the bidding phase we have to keep in mind that the resulting contact will be played and that any bid gives away information about the hand to the partner and the opponents, which can be used during the playing phase. During the playing phase on the other hand, we have to play based on the contract while exploiting information obtained from the bidding. More detailed explanation of the rules of bridge is available in the appendix.

[1]Contract: The main result of the bidding. It determines the declarer, the trump suit (or no-trump) and the tricks the declarer needs to score in order to gain a positive score. It also sets the correspondence between the tricks scored and the outcome.

[2]Trick: The basic form of scoring. Once every four cards played, the partrenrship who played the trump of the highest rank or the card of the highest rank of the suit led wins one trick. There is a total of 13 tricks.

An interesting simplified version of bridge, called double-dummy bridge, is also important for this thesis. This can be considered as the perfect information variant of bridge. In this game each pair can be considered as a player (in real bridge this is not feasible since each player has different information), and tries to maximize the number of tricks taken. It is played exactly like the playing phase, with one noteable exception. All the cards are revealed to all the players. Each player has to find the optimal strategy based on each players' cards, the trump suit and the player that leads the first trick. Human players use double-dummy bridge in order to analyse games. It almost never has any usefulness during human play. However, in computer bridge, it is commonly used as the cornerstone of bridge playing agents. It is commonly used during both the bidding and the playing phase. Although it is not the only part of a computer bridge agent, a double-dummy solver is arguably the most critical one.

Bridge is widely considered as the card game equivalent to chess. It is actually a mental sport, just like chess. This is the reason that humans strive to constantly improve as bridge players and gain prestige in the process. This also urges computer scientists and engineers to build better automated bridge players. A computer bridge competition also exists. The above statements indicate that the game of bridge is interesting as well as challenging. The challenge of computer bridge can further be reinforced by the fact that no computer bridge player has achieved a victory against a top human team.

There are a lot of factors that make bridge intersting and challenging, for both humans and computers. Below we are going to enumerate some:

1. Even the simplified perfect information version of bridge, double-dummy bridge is non-trivial.

2. It being an imperfect information game, the player has to play based on propabilities.

3. Potential difficulties later on in the play have to be taken into account (i.e. there is an important sequential aspect in the decision making process).

4. Each player tries to convey useful inforation to their partner, without giving away useful information to the opponents. These two objectives are usually mutually exclusive.

5. During bidding, it is usually difficult to communicate effectively with our partner, because the opponents' bidding interferes with our bidding and disrupts the comunication between us and the partner.

6. During the bidding, too much interference with the opponents might lead to a disaster, but too little interference might allow the opponents to reach the optimal contract easily. A balance has to be found about how much we should interfere. Finding such a balance during the bidding is extremely hard.

7. Parts of the game (primarily the bidding and secondarily the defence) are heavily based on partnership agreements. Some agreements are mutually exlusive (for example an opening bid of 2♣ can mean many things, but the partnership has to choose only one or two meanings). This means that different partnerships use different agreements, while there is no known agreement that can dominate all others.

Bellow we are going to briefly describe the challenges in building an automated bridge player:

1. A fast double-dummy solver has to be implemented, that can be used as a basic component of the agent.

2. Exhaustive exploration of each possible scenario is impossible, since there are too many. Humans tend to use intuition to group possible scenarios, but current computer bridge players lack such intuition. However, recent research in the poker-playing domain might help in this respect[3, 4].

3. Lack of information has to be considered not only in the current move, but in the next moves as well. Humans can intuitively detect future difficulties just by looking at the cards, but computers cannot.

4. Since it is a four-player game, multi-agent practices need to be deployed. Sometimes treating the game as two player game is a good abstraction, however this is not the case for the bidding phase.

5. The bidding phase is hard in a game-theoretic sense. The fact that each bid conveys information makes things even harder, since a bid can have unexpected results after the bidding is over.

6. An infrastructure has to be implemented, in order to "understand" and play according to different agreements.

7. Sometimes humans deviate from the agreements (the hand might not match with the agreed meaning of the calls made), when they think that the benefists of confusing the opponents outweight the drawbacks of confusing the partner. Sometimes they deviate from the agreement, because they think that the agreements will lead to a bad contract.

In order to implement our agent, we have to overcome the above difficulties. They are related to the lack information, some game-theoretic aspects, as well as the computational complexity of the double-dummy problem.

1.1 The Game Theoretic Aspect of Bridge

Bridge is played by two opposing pairs. Each player has a partner and two opponents. It is cooperative game between the partners and competitive between the opponents.

Bridge is a zero-sum game[2], which means that the sum of scores of the teams is zero. Because in a bridge table there are two pairs, the score of one pair equals the opossite of the score of the other pair. Other zero-sum games are chess, backgammon, Go and poker.

It is an imperfect information game[2], because not all cards visible to every player.

It is a complete information game[2], every player's objective is known to everyone.

1.2 Thesis Outline and Contributions

Initially, we implemented a program that solves the double-dummy bridge problem. It is a computationally demanding problem, not solvable by a naive applicaton of the minimax algorithm. However, it can be solved fast enough, by using optimizations based on specific aspects of the problem. A significant part of this thesis is commited to explaining those aspects, as well as the implementation of the solver in particular. We also proved theoretically that the optimizations used do not give wrong solutions.

Then, we implemented and tested various methods of tackling the lack of information of the playing phase of the single-dummy bridge. The playing phase of bridge is played by three players, the declarer and the two defenders. It is an imperfect information, zero sum game, in which the two defenders work as a team. The lack of information affects mostly the declarer, and for that reason we worked mostly on declarer play. The effect of using a defender that conceales information is also tested.

The bidding phase is the other phase of bridge. It is the most decisive phase in human bridge games, as well as the most challenging part for a computer. It can be modelled as a signalling game[5]. In this thesis we are not going to implement a complete bidder, because it far surpasses the level of a diploma thesis.

In this thesis, we devised some methods for solving the double dummy problem, and implemented them in practice. We also proposed a novel, more strategic method of dealing with imperfect information during the playing phase, especially from the point of view of the declarer. This method, contrary to the Monte-Carlo method, takes into account potential lack of information after the current move. We also optimized this method and mathematically proved that the optimization does not affect the selected line of play. We also proposed an improvement that is expected to further improve the quality of the agent, under certain circumstances. We have implemented three agents, one based on the Monte-Carlo method, one based on our novel method without the improvement, and one based on the novel method with the improvement. All three agents make use of the double-dummy solver. We also implemented two perfect information defenders. One that selects an arbitrary good move and one that selects, out of the good moves, the one that is more likely to make the declarer err.

Finally we experimentally compare the outcomes of each of the three declarers and the two defenders over a large number of deals, in order to back our claims.

The core of the thesis consists of the following chapters:

1. Solving the double dummy problem: In this chapter, we define the double-dummy problem, as well as the algorithm we used to solve it. The optimizations used as well as the necessary proofs that the optimizations do not affect the solution are also provided.

2. Dealing with imperfect information: In this chapter, we show the importance of information in single dummy bridge. We also describe the most widely used method of tackling information in single-dummy bridge, the Monte-Carlo method. We also describe the novel method we implemented, show why it is expected to work better than the Monte-Carlo method, and some optimizations and improvements. We also propose methods for the defenders to exploit imperfect information. This chapter is the most constructive in the thesis, since it addresses a novel approach.

3. Testing our agent: This chapter shows how the different methods where tested against each other, the significance of the results and a review of the methods based

on the results. This chapter shows in practice the advantages and the drawbacks of the novel methods implemented.

Chapter 2

BACKGROUND

Automated game playing, has been a field of artificial inteligence reasearch since long ago. The idea of computers playing games like chess was first intruduced by Shanon in 1949 and the first known programs were implemented in 1956 for checkers and 1957 for chess. The first automated game players played solely perfect information strategic games, like chess. Nowadays computers beat humans against theses type of games, with the most renowned victories of artificial inteligence being Deep Blue (1997) and AlphaGo (2016).

On the other hand reasearch on imperfect information games is not so well-developed. Such games are most card games like bridge and poker. Other board games also fall into this category. In the majority of those games computers do not yet play as well as human experts. An automated player has to manage the lack of information, in addition to the problems existing in perfect information games. Information gathering and managing imperfect information is where human players still outmatch computer players.

However, recent research in poker shows that we are getting closer to building automated agents that can beat top humans in imperfect information games. In fact, heads-up limit holdem, a variation of poker, was reported to be essentially weakly solved by Michael Bowling, Neil Burch, Michael Johanson and Oskari Tammelin in January 2015[7]. Both poker and bridge are imperfect information games. There are cases in bridge, albeit uncommon, where poker tactics work. However, those two games are completely different. An important difference between poker and bridge is that in poker each player plays on his own, while in bridge each player has a partner. This adds the aspect of communication in bridge and renders bluffs almost always ineffective, since they can confuse the partner. Another important difference is the way those games are played. In poker the goal is to earn as many chips as possible or to lose as little as possible by betting, folding, etc. In bridge on the other hand, the goal is reaching the optimal contract and finding a strategy that scores the maximum number of tricks. Although great differences between poker and bridge exist, computer bridge might benefit from research on computer poker, since some poker tactics sometimes work in bridge[16].

A method commonly used in automated game playing is the Monte-Carlo method. Monte-Carlo tree search is used mostly in perfect information games like Go[8]. It selects the move by randomly expanding nodes until the end of the game is reached and repeating this process a large number of times, to obtain statistical information. In many imperfect information games, including bridge, the Monte-Carlo sampling aproach is used. In bridge, for instance, a large number of potential layouts of cards is generated and each layout is solved independently. The best move is then selected according to the average payoff over all samples.

The rest of this chapter is dedicated to computer bridge, including a brief description of past research on the subject. We are going to describe a brief history of computer bridge, research on double-dummy bridge, the playing phase and the bidding phase. We are finally going to present some other aproaches to the subject.

Computer bridge was played at the first Computer Olympiad events taking place since 1989. Of course, the first automated bridge player was built before that year. Since 1997 the World Computer Bridge Championship is jointly organized by the American Contract Bridge League and the World Bridge Federation (until 1999 the event was organized solely by ACBL). As expected, the first automated bridge players were pretty weak, but they progressively got stronger. The first player that could play reasonably well agaist a human was GIB[12], which earned the World Computer Bridge Championship in 1998. Nowadays, automated bridge players can compete against human experts. However, an automated bridge player that can undisputedly defeat the top human team, has not yet been built.

The double-dummy problem has been researched quite well in the past years. The first solvers were based solely on conventional minimax search methods, like α-β pruning and transposition tables. However, those methods were not fast enough to be used in practice. In 1996, Ming-Sheng Chang, proposed methods of using specific properties of double-dummy bridge, in order to optimize the search algorithm for NT (no-trump) contracts[9]. This method, albeit much faster than the generic minimax methods, was still not fast enough. The first fast double-dummy solver was built by Ginsberg and it was used in GIB[12]. It deployed the partition search method, which gave a speed-up comparable to the α-β pruning[12]. Curently, the double-dummy problem can be solved in tractable time. The best, publicly available solver is the DDS library, by Bo Haglund and Soren Hein, which can solve several problems in a few seconds[13].

The playing phase of bridge has been researched extensively, especially for declarer plays. FINESSE, one of the first automated bridge players, played based on tactics used by humans instead of cards. When fast double-dummy were built, the Monte-Carlo sampling aproach was used. This method selected the card that gave the highest expected score over a randomly generated set of double-dummy problems. This was first used by GIB, achieving pretty good results. However, it could not gather information well enough. The methods of Vector Minimaxing and Payoff Reduction Minimaxing have been proposed and tested. The results were quite satisfactory, but the high complexity made those methods unusuable in practice[14]. On the other hand, the method of Achievable Sets, deployed in GIB, could be used in practice and gave good results. GIB achieved a good place in a declarer card play competion among human experts[12]. However, the method of achievable sets is only "interested" in making the contract, completely ignoring overtricks or undertricks.

The bidding phase on the other hand, has no satisfactory solution yet. Because of the ambiguity of each bid, the bidding phase is a much more challenging aspect of bridge. Older automated bidders decided each call by blindly following human created conventions (only one option could meet the requirements stated by the conventions). Later, the double-dummy solver was used for the bidding phase, in a similar manner it was used in the playing phase, in order to implement Monte-Carlo bidders. Those bidders are better than the older ones, because they do not have the constraint of one possible call. The drawbacks of the Monte-Carlo sampling aproach are much stronger in the bidding phase. Such bidder was used by GIB[12]. Some promising research on bidding was done by Amit and Markovitch in 2006 using machine learning techniques.

This method builds a stronger bidder than the one used by GIB after enough learning[15].

Other methods used in Computer Bridge implement artificial neural networks and machine learning. Most research currently taking place is about those. It can be used in all three above-mentioned aspects, but the error rate is significant.

Chapter 3

THE RULES OF BRIDGE

The basics and some definitions

Bridge is a card game played by two pairs of two players. The players are often named N, E, S and W, after the four cardinal directions. Each deal begins with the cards being dealt to each player. Each player receives 13 cards. Then the bidding phase starts which determines the contract and the declarer. Then, the playing phase ensues. Finally, the outcome of the deal is calculated.

We are now going to define some useful terminology about bridge. We are only interested in terms that affect the rules of bridge, not the terms that define bidding and playing strategies.

Auction Another name for the bidding phase.

Declarer One player of the pair that bids highest during the auction. The way the declarer is determined and the effect at the rest of the deal will be thoroughly explained later.

Dummy The declarer's partner.

Defenders The opponents of the declarer.

Trump A suit that is decided during the auction. It affects the playing phase greatly.

Stain Any of the suits or No-Trump(NT). The five stains are ♣, ♢, ♡, ♠ and NT.

Trick Each deal yields a total of 13 tricks, one for every 4 cards. Each pair's goal during the playing phase is to maximize the number of tricks taken.

3.1 The Bidding Phase

The bidding phase, also known as the auction phase, begins after the cards are dealt. The outcome of the bidding is called the contract and greatly affects the rest of the game.

The bidding is done clockwise starting with the dealer. When a player has his turn to call, they can say "pass" or a number between 1 and 7 (called level) and a stain (we will now call this a bid). Example bids are 1♠, 3♢, 5NT and 7♣. However, not all bids are allowed. After the first bid (which allows any number and any stain), each number-stain bid has to be stronger than the previous one. A bid is stronger than another bid if the

number is higher, or if the number is equal and the stain is stronger. The strength of stains is ♣, ♢, ♡, ♠, NT in ascending order. This means that after a 2♡ bid, 3♣, 2NT, 2♠ and 6NT are valid bids, but 1NT, 2♣ and 1♢ are invalid bids. "Pass" is always a valid call.

If the last non-pass call was a bid by the opponents, the player has the choice to call "double". If the last non-pass call was an opponents "double", the player has the choice to call "redouble". "Double" and "redouble" have an effect to the final score. That effect is canceled if another bid is made after the "double" or the "redouble", and "double" can be called again.

The bidding phase ends after 3 consecutive "pass" calls. The only exception to this rule is when the auction starts with 3 "pass" calls, in which case the auction continues if the fourth player bids. If all four players pass, the deal ends prematurely and both pairs score 0 points. In any other case, a contract is decided. The contract is redoubled if the last non-pass call was "redouble" and doubled if the last non-pass call was "double". Otherwise, the contract is undoubled. Part of the contract is also the level and the stain of the last bid. The declarer is the first player, of the side who made the last bid, that bid first in the contract stain. If the contract stain is NT, the deal will be played without trumps. Otherwise, the trump will be the contract stain. The contract is usually notated by a few characters that show the level, the stain, the declarer and the indoubled/doubled/redoubled info. By that notation 3NTE means that the contract is undoubled, the level is 3, the stain is no-trump and the declarer is East. 4♠Nx means that the contract is doubled, the level is 4, the trumps are the spades and the declarer is North. 7♢ Wxx means that the contract is redoubled, the level is 7, the trumps are the diamonds and the declarer is West.

Bellow are some bidding examples, and the contract resulting from them. P means "pass", X means "double" and XX means "redouble".

W N E S 1♠ P P	Incomplete auction. If West passes, the contract will be 1♠N.
W N E S 1♠ P P X P P P	1♠Nx
W N E S 1♠ P 2♡ P P P	2♡S
W N E S 1♠ P 2♠ P P P	2♠N
W N E S 1♠ P 2♠ P P X XX P P P	2♠Nxx
W N E S 1♠ P 1NT P P P	1NTS
W N E S 1♠ P 1♡ P P P	Invalid bid. 1♡ is not stronger than 1♠.
W N E S 1♠ P X P P P	Invalid "double". Can only double an opponent's bid.
W N E S 1♠ X 2♡ P P P	2♡ S
W N E S 1♠ 2♠ P P P	2♠E
W N E S P P P P	All players passed, the deals ends with 0 points for both sides.

3.2 The Playing Phase

After the auction is over and the contract is decided, it is time for the playing phase. The playing phase also takes place clockwise. The declarer and the trump suit (or its non-existence) are part of the contract. The player at the declarer's left hand plays a card. It has to be any of the 13 cards in his hand.

Then the dummy - the declarer's partner - puts his hand face up on the table and his role in the deal is over. All 13 dummy's cards are now visible to the players, but are not yet played. From now on each defender will play his hand and the declarer will play his hand and the dummy hand. Every four cards played (one from each player) make a trick. The first trick begins with the card played before the dummy shows his hand. When a hand is played, it is shown to everyone and gets removed from the player's hand. The first card of every trick can be any card of the players hand. The other three players

have to play a card that matches the suit of the leading card of the trick. If they cannot follow suit, because they have no cards of that suit left, they can play any card. When all four players have played, the winner of the trick is decided. If at least one player has played a card of the trump suit, the player who played the highest trump wins the trick. Otherwise, the player who played the highest card of the suit of the first card wins the trick. The player who won the trick leads the next trick. This process continues until all cards are played.

Below are some examples of tricks. All those tricks are led by West and the trumps are spades. This is just an assumption to make these examples easier to understand.

W	N	E	S	
3♠	7♠	A♠	2♠	East wins the trick by playing the highest trump.
W	N	E	S	North wins the trick by playing the highest trump. East has no spades.
3♠	7♠	A♢	2♠	
W	N	E	S	East wins the trick by playing the highest card in the suit led (diamonds).
3♢	7♢	A♢	2♢	
W	N	E	S	South wins the trick by playing the highest trump. South has no diamonds.
3♢	7♢	A♢	2♠	
W	N	E	S	West wins the trick by playing the highest card in the suit led (diamonds). North, East and South have no diamonds.
3♢	7♣	A♡	2♡	

The contract is important duting the playing phase, because it determines the trump suit (or the non-existence of trumps), the declarer and the number of tricks required by the declarer for a positive score.

When all cards have been played, we count the tricks that each pair won. The sum of the tricks each pair has won is always 13. The number of tricks and the contract determine the final score of the deal.

3.3 Calculating the Score

After the deal has been played, we have to calculate the score. The score depends on the contract and the number of tricks the declarer's side has won. It also depends on the declarer's vulnerability, a status determined and made known to the players before the cards are dealt. It is usually determined by the serial number of the board (no pair is vulnerable during the first board, N-S pair is vulnerable during the second board, E-W pair is vulnerable during the third board and both pairs are vulnerable during the fourth board). The sum of the declarer's and the defenders' score is always zero. We will explain the calculation of the score for the declarer. The defenders will have the opposite score.

The declarer's score is positive when the tricks won is equal or higher to the contract level increased by 6. Otherwise it is negative. For example 3NT will give a positive score to the declarer if they win at least 9 tricks and 7♣ will give a positive score if they win all 13 tricks. The outcome of the deal is often notated like "3NTN+1", "2♢S-2" and "6♠E=". The first part of this notation is the contract, and the later part shows us the number of tricks won compared to the needed tricks. In these examples, the declarer won 10, 6 and 12 tricks respectively. When the declarer wins less tricks than needed, the

missing tricks are called undertricks. On the other hand, when the declarer wins more tricks than needed, the excess tricks are called overtricks.

Bellow, we are going to explain the penalty when the declarer has undertricks. In an undoubled contract, each undertrick will cost the declarer 50 points when not vulnerable and 100 when vulnerable. When the contract is doubled and the declarer is not vulnerable, the penalty is 100 for the first undertrick, 200 for the second and the third and 300 for each further undertrick. When vulnerable, the penalty is 200 for the first undertrick and 300 for each further undertrick. When the contract is redoubled, the penalty is twice the penalty of a doubled contract. When the declarer has undertricks, no other score is added to the penalty.

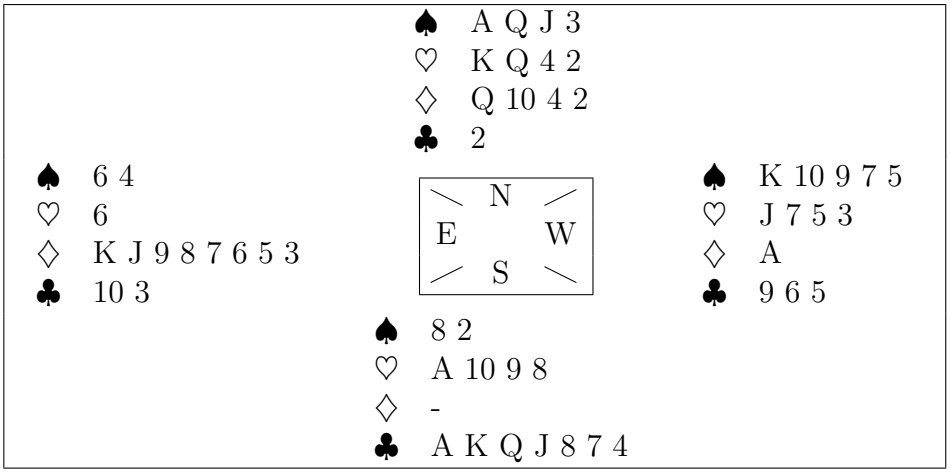
When the declarer makes the contract, the score calculation is more complex. First, we have the points the contract is worth. A ♣ and a ♢ contract is worth the contract level multiplied by 20, a ♡ and a ♠ contract is worth the contract level multiplied by 30 and a NT contract is worth the contract level multiplied by 30 added by 10. A doubled contract is worth twice the undoubled one and a redoubled contract is worth four times the undoubled one. We also have the overtrick bonus which is 20 for each overtrick in an undoubled ♣ or ♢ contract, 30 for each overtrick in an undoubled ♡, ♠ or NT contract, 100 for each overtrick in a doubled contract when not vulnerable, 200 for each overtrick in a doubled contract when vulnerable or a redoubled contract when not vulnerable and 400 for each overtrick in a doubled contract when vulnerable. The declarer also receives 50 points partscore bonus if the contract is worth less than 100 points, but they receive the game bonus if the contract is worth 100 or more points. The game bonus is 300 points when not vulnerable and 500 when vulnerable. When the contract level is 6, the small slam bonus applies, which is 500 when not vulnerable and 750 when vulnerable. When the contract level is 7, the grand slam bonus applies, which is 1000 when not vulnerable and 1500 when vulnerable. Finally, if the contract is doubled or redoubled, the declarer receives 50 or 100 points respectively as "insult" bonus.

Bellow we show some examples of contracts and results. We assume that NS is vulnerable, while EW is not.

Contract	Tricks	Result	Score
1NTE	7	1NTE=	90
1NTN	7	1NTN=	90
3NTE	9	3NTE=	400
3NTN	9	3NTN=	600
3NTE	8	3NTE-1	-50
3NTN	8	3NTN-1	-100
4♣E	10	4♣E=	130
3♣E	10	3♣E+1	130
6♣E	12	6♣E=	920
5♣E	12	5♣E+1	420
5♣Exx	12	5♣Exx+1	1080
7♣E	12	7♣E-1	-50
7♣E	13	7♣E=	1540
7♢Nx	6	7♢Nx-7	-2000
7♢Nx	13	7♢Nx=	2330
7♢Nxx	13	7♢Nxx=	2660
2♠Nx	8	2♠Nx=	670

3.4 An Example Deal

In the following deal, we will show how the bidding phase can go, as well as the playing phase. The following cards were dealt, with North being the dealer and N-S vulnurable.



The bidding went as follows:

North opened the auction with 1♢ showing at least four diamonds (rarely three) and a relatively strong hand. East bid 1♠ showing at least five spades and South doubled showing at least four hearts. The meaning of that double was an agreed between North and South. West passed and North bid 3♡ showing at least four hearts and a strong hands. This also established hearts as the desired trump suit. East passed and South bid 4♣ showing that she either has the A♣ or no clubs. West passed and North bid 4♠ showing the A♠ or no spades. East passed and South bid 5NT. That was a conventional bid that asks partner to bid 7♡ if she holds two out of the three highest hearts and 6♡ otherwise. West passed, North (holding the K♡ and the Q♡) bid 7♡, followed by three conscutive passes. The contract of 7♡N was reached. A summary of the auction is given below:

W	N	E	S
	1♢	1♠	Double
Pass	3♡	Pass	4♣
Pass	4♠	Pass	5NT
Pass	7♡	Pass	Pass
Pass			

The following table shows the playing phase. South's (dummy's) hand was revealed to everyone after A♢ was played by East. It is important to note that each card was played exactly once. The underlined card, is the one that won the trick.

W	N	E	S
9♢	2♢	A♢	8♡
6♡	2♡	7♡	A♡
3♢	K♡	3♡	10♡
10♣	2♣	9♣	A♣
3♣	4♢	6♣	K♣
6♢	10♢	5♣	Q♣
4♠	3♠	5♡	J♣
6♠	Q♡	J♡	9♡
J♢	A♠	5♠	2♠
7♢	Q♠	10♠	8♠
8♢	J♠	K♠	4♣
5♢	4♡	9♠	7♣
K♢	Q♢	7♠	8♣

The declarer scored 10 tricks, while the defenders scored three tricks. The result was 7♡N-3, yielding a score of -300 to N-S and +300 to E-W.

Part II

DESIGN AND IMPLEMENTATION OF THE AGENT

Chapter 4

SOLVING THE DOUBLE-DUMMY PROBLEM

4.1 The double-dummy problem

Double-dummy bridge is a simple variation of bridge. It is a perfect information, competetitive, two player game. In that sense it is similar to chess, checkers, tic-tac-toe and reversi, with different search tree size.

Definition 1. The game of double-dummy bridge:

- A card deck of 52 cards is dealt to four hands, each hand receiving 13 cards. The four hands are usually named after the cardinal directions.
- The cards of each hand are visible to each player. There are no hidden cards.
- There are two opposing players, one playing the cards of the North and South hands, while the other playing the cards of the East and West hand.
- The player who makes the openning lead, as well as the trump suit (or NT) are selected before the first card being played. They may be considered as parameters of the game (like the distribution of the cards).
- The game is played according to the rules of bridge. A move is a selection of a card that can be played from the hand that has to play.
- Each player's goal is to maximize his number of tricks.

The key differences between double-dummy bridge and single-dummy bridge are the following:

- There is no bidding phase. The stain of the contract (a trump suit or NT) as well as the player who plays first are given at a double-dummy problem.
- It is a perfect information game. Every card of each player is known to every player.
- Each pair of hands can be considered as a single player. This makes double-dummy bridge a two-player game.

- There is no contract level[1], since it cannot affect the cards played. Maximizing the number of tricks will maximize the score for any contract level.

Double-dummy is a constant-sum minimax game. It can be easily converted into a zero-sum game, by subtracting tricks of the minimizer from the tricks of the maximizer.

Definition 2. The double-dummy problem:
The problem of finding the optimal moves of a player and/or the corresponding minimax score in the game of double-dummy bridge. The game may be in the initial state (no cards have yet been played) or a later state. In the case of a later state, the score might be, either the total tricks expected to be won, or the remaining tricks expected to be won.

The double-dummy problem is computationally demanding, and cannot be solved in a tractable time without some major optimizations. The optimizations we used include:

- α-β pruning
- transposition table
- iterative deepening zero window search
- partition search
- move ordering
- forward pruning through the use of admissible heuristics

In the following sections we are going to explain the above optimizations. We are also going to prove that the solution is always correct, provided that the implementation has no bugs.

4.2 The basic solver

Initially we implemented a simple minimax solver. As score we define the difference of the players' tricks. With this score it is a zero-sum game. The only irregularity of this game, is that one player can play twice in a row. This happens when the player who led a trick loses it. This makes the implementation of algorithms like negamax and negascout more complex[10], but the algorithm that we use does not get affected.

We then implement α-β pruning. This is an optimization applicable to any minimax problem, giving a great performance boost. This optimization does not affect the outcome of the minimax algorithm, only its efficiency[11].

[1]Contract level: A number between 1 and 7 that determines the number of tricks the declarer needs in order gain a positive score in a contract. The tricks needed equal the contract level added by 6. For greater detail about the effect of the contract level in scoring see apendix A

4.3 Implementing and optimizing the transposition table

The use of a transposition table is another common optimization to the minimax algorithm, mainly used in chess engines. It is used to store the results of nodes of the search tree that we have already searched. This makes the search faster by pruning the nodes the results of which have been discovered to cause an α or a β cutoff. In our transposition table we store the upper and the lower bound of the score achievable at the **remaining** tricks, not the total tricks. Using the total tricks would make the transposition table far less efficient, because of the nature of double-dummy bridge. This will be further explained below.

4.3.1 The naive implementation

We first need to define when two positions are equal.

Definition 3. Two positions are equal when:

1. Each hand has the same card in both positions. In the corresponding double-dummy problem this is true if and only if the same card have been played.
2. If we are examining two positions at the beginning of the trick, the leading hand of the trick should be the same.
3. If we are not examining two positions at the beginning of the trick, the suit lead and the winning card should be the same.

It should be noted, that if the first condition holds true, both positions have the same number of cards played at this trick.

If both players play according to the minimax algorithm, they will win the same remaining tricks out of two equal positions. However, there may be a difference in the tricks won until they reached the equal positions. The total tricks will be the tricks so far, added to the remaining obtainable tricks. For example, let's assume that the search algorithm reaches a position P with a score of 4 and at the remaining tricks the score is 1. This will mean that the total score is $4 + 1 = 5$. Later at the search, the algorithm reaches the same position P, by having a score of 2. This will mean that the total score will be $2 + 1 = 3$.

In the naive transposition table implementation, we implement a hash table that has equal keys for equal positions. If the search algorithm reaches a position searched before, it will retrieve the upper and the lower bound of the score of that position, through the hash table. If we stored the total score at the transposition table, we would have an extra condition for equal positions. The positions should have the same score so far in order to be equal. That would make hits at the hash table more uncommon, negatively affecting the performance.

4.3.2 Implementation of the partition search

Equal positions have always the same minimax score for the remaining tricks. However, it is not necessary for two positions of the same scores to be equal. The term "equivalent position" is useful in this case.

Definition 4. Two positions are equivalent when they have equal minimax score for the remaining tricks.

Using the same keys in the transposition table for equivalent positions can greatly improve the performance of the search algorithm. This method was first used in the implementation of GIB, an old computer bridge champion, and it was named partition search[12]. In order to use partition search, we have to distinguish equivalent positions. Finding all equivalent positions of a certain position, is harder than the double-dummy problem itself (in fact, it requires solving the double-dummy problem), so it is out of the question. However, finding a subset of equivalent positions, is computationally easy enough to give us an impressive performance boost.

In our implementation, we distinguish equivalent positions, based on the ordering of the cards. We first need to define the relative rank of a card, the relative hand and the relative position.

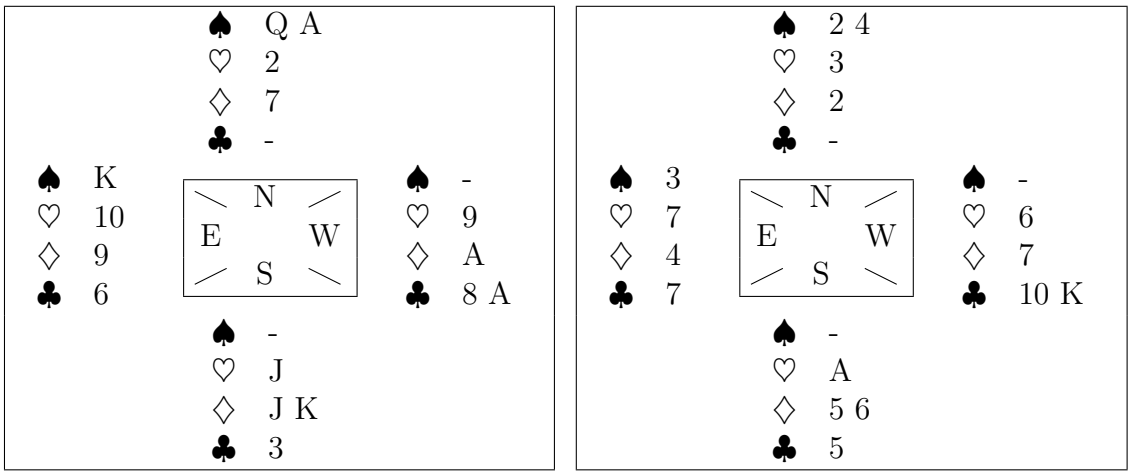
Definition 5.

Relative rank The relative rank of a card C is defined as the number of unplayed cards, that have the same suit as C and are outranked by C.

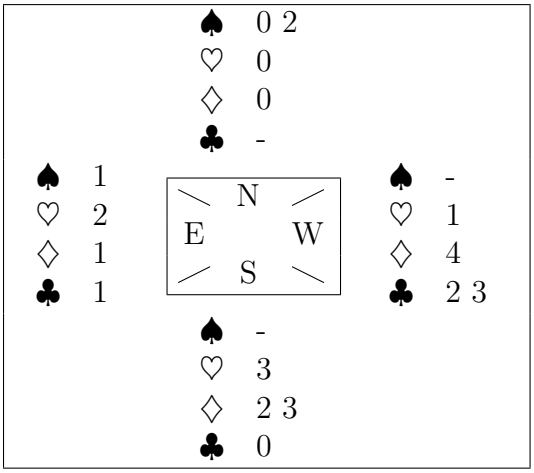
Relative hand The relative hand is defined as the relative rank of the cards of the hand, grouped by suit.

Relative position The relative position is defined as the four relative hands of the position.

Below is an example displaying the relative position of two positions:



The relative position of the above positions, which is the same for both positions, is given below:



In the above figure, 0 means the lowest unplayed card in a specific suit, 1 the second lowest and so on. For instance, in the above examples South has no spades, the fourth lowest unplayed heart (relative rank 3), the third and fourth lowest unplayed diamond (relative ranks 2 and 3) and the lowest unplayed club (relative rank 0).

The above positions are equivalent under some conditions. In general, we use the theorem below to generate a subset of equivalent positions.

Theorem 1. *If two positions*

- *Have the same relative position.*
- *Have the same hand leading the trick. If the positions are not at the start of the trick and the above statement holds true, the second statement is always true.*
- *The suit lead and the positions have the same relative position after adding the current trick winning card to the hand that played it, provided that the positions are not at the beginning of a trick.*

Then the positions are equivalent.

Proof. Two positions cannot satisfy the first condition, unless they have the same number of unplayed cards. Thus we can only assume positions that have the same number of unplayed cards.

Firstly, every position with 0 unplayed cards has a minimax score of 0, as there are no tricks left. This means that all positions with 0 unplayed cards are equivalent. Now we consider that any two positions that satisfy the above conditions and have $n-1$ unplayed cards are equivalent, where $1 <= n <= 52$. The theorem will be proved if we prove that the same holds for any position of n unplayed cards.

If n is divisible by 4, which only happens when the positions are at the beginning of the trick:

The hand that leads the trick has their turn to play. It has to be the same hand for both positions because of the second condition. Since the leading hand has the same relative hand in both positions, for each card playable in the first position, there is one playable card at the second position of the same suit and the same relative rank. If both positions play a card of the same suit and the same relative rank, the resulting positions will

1. Have the same relative position.

2. Have the same suit lead.

3. Have the same relative position after adding the current trick winning card to the hand that played it. This will be the relative position of the initial positions.

This means that for each move in the first position (including the optimal move which is what the minimax algorithm would choose) there is a move in the second position that makes both successor positions, which have $n-1$ unplayed cards satisfy the requirements and are considered equivalent.

If n in not divisible by 4, which only happens when the positions are not at the beginning of the trick:

By comparing the relative positions of the current positions and the ones with the added winning card we can find out the suit of the winning card and the hand that played it. We can also find out the cards, in the hand that is about to play, that can outrank it. When the positions satisfy the above restrictions, the number of cards in the hand that plays that can beat the current winner are equal and they are the highest ranking hands of the hand in a certain suit. This means that if two cards from two positions that satisfy the restrictions have the same suit and the same relative rank, they will either both or neither beat the winner. In either case, the successor positions (with $n-1$ unplayed cards) will satisfy the restrictions. The fact that the suit led is the same in both positions, assures that if a card is playable in one position its equivalent will playable in the other position. This proves that for each move in the first position, there is a move in the second one that makes both successor positions equal.

The above, by means of mathematical induction, prove our theorem. □

4.3.3 Implementation details about the transposition table

In our hash table the key consist of two parts, the hash (or index) and the tag. We have implemented functions that return a key of any position. Two equal position position will always return equal keys and two equal keys will only be returned by equivalent positions. A 27-bit hash and 64-bit tag can generate enough keys, not only for all positions that can appear in a double-dummy problem, but for all positions that can appear to all double-dummy problems when two hands are constant. We only treat the positions that satisfy the above criteria as equivalent for the transposition table. The information about the position that is present in the key is:

1. 4 numbers showing the relative position, one for each suit.

2. two bits showing the player to play (only for the beginning of the trick).

3. two bits showing the suit led (not for the beginning of the trick).

4. one bit showing if the winning card is a trump (not for the beginning of the trick).

5. one number showing the relative position considering the winner as unplayed (not for the beginning of the trick).

We compute the relative positions before we start solving the double-dummy problem. We also shuffle the above bits, in order to prevent hash collisions as much as possible. As

data to each key we store the minimum and the maximum score of the location, as well the depth of the search that the scores represent. The depth is necessary in optimizations that will be explained bellow.

4.4 Implementing an iterative deepening zero window search algorithm

In double-dummy bridge, each trick has a permanent effect at the outcome of the game. If a trick is lost the maximum number of tricks obtainable decreases by one and if a trick is won the minimum number of tricks that can be lost decreases by one. This property of double dummy bridge, allows us to use a zero window minimax algorithm, where the depth is the number of tricks (sets of four moves that decide a trick, to be exact) that we are searching.

4.4.1 The primitives of the idzws algorithm

Double-dummy bridge has a property that most minimaxing problems do not. Since the score of the game is tied to the total number of tricks won, winning or losing a trick has permanent effect to the outcome of the game. Winning or losing a trick is decided once every four moves.

Initially, we modify the minimax algorithm by adding a depth argument. If the depth is N we only examine the N first tricks, and the score will be the difference of the tricks won by the two opposing players. The score S can be $-N \leq S \leq N$ and S is odd if and only if N is odd. If the depth is 13, this algorithm will give the results of the original algorithm. If the depth is 0, the score is always 0, since neither player can win any tricks.

Theorem 2. *If the minimax score of a double-dummy problem at depth N ($0 \leq N < 13$) is S, at depth $N+1$ the minimax score will either be $S+1$ or $S-1$*

Proof. Let's define as M the tricks won by the maximizer and m the tricks won by the minimizer. A score of S at depth N, means that $M - m = S$ and $M + m = N$, meaning that $M = (N + S)/2$ and $m = (N - S)/2$.

At depth $N+1$, the total number of tricks is increased by one, meaning that $M'+m' = M + m + 1$. However, due to the nature of the double-dummy problem, $M' \geq M$ and $m' \geq m$. This means that either $M' = M$, $m' = m+1$ and $S' = M' - m' = M - m - 1 = S - 1$ or $M' = M + 1$, $m' = m$ and $S' = M' - m' = M + 1 - m = S + 1$. □

This allows us to compute the minimax value of a position using the following algorithm:

Algorithm 1. Supposing that minimax(a, b, N) computes the minimax score of the problem at depth N, with α cutoff being a, and β cutoff being b. We use the following algorithm to compute the minmax score at depth 13.

```
score = 0;
for(i = 1; i ≤ 13; i++) score = minimax(score-1, score+1, i);
return score;
```

This is an iterative deepening algorithm, since it begins solving the problem at depth 1 and the depth increases by one each step, until it reaches the required depth. It is a zero-window search algorithm, because each search of minimax can only return two possible values, score-1 and score+1. Because these values are the cutoff boundaries and there is no value between them that can be returned, α and β cutoffs are much more common. Because of the above theorem, the algorithm returns the correct score.

4.4.2 Using the above algorithm with a transposition table

In order to use the idzws algorithm with a transposition table, we need to store the search depth at the transposition table. It can be easily proven that the minimum score at depth M, provided that the minimum score at depth N, $N \leq M$ is S, is $S + N - M$. If the same information concerns the maximum scores, it would be $S - N + M$. However, when reusing the transposition table after solving another double dummy problem, the depth stored at the transposition table can be bigger than the current search depth. We will prove that the idzws algorithm can use the information stored at the table without failing the final outcome. Proving this however is no simple matter.

We first need some definitions:

Definition 6. Terms used in the generic idzws algorithm:

$MM(N)$ The minimax score of the double-dummy problem at depth N. $MM(13)$ is the minimax value of the complete game.

$C(N)$ The set of scores that satisfy the following restriction:

$x \in C(N) \Leftrightarrow |x - MM(13)| \leq (13 - N), (x - MM(13)) \equiv (13 - N) \mod 2.$

$C(N)$ is defined as the consistent score set of a double-dummy problem at depth N.

We will now show some properties of consistent score sets.

Theorem 3. *properties of* $C(N)$*:*

1. $MM(n) \in C(n) \forall n, 0 \leq n \leq 13$
2. $C(13) = \{MM(13)\}$
3. *if* $0 \leq m \leq n \leq 13$ *and* $x \in C(n)$*, then* $\forall y \in [x + m - n, x - m + n]$ *and* $(x - y) \equiv (m - n) \mod 2 \Rightarrow y \in C(m)$
4. *if* $0 \leq n \leq m \leq 13$ *and* $x \in C(n)$*, then* $\exists y \in [x - m + n, x + m - n] \Rightarrow y \in C(m)$

Proof. 1. The nature of the double-dummy problem proves it.

2. Assuming $x \in C(13)$ then $|x - MM(13)| \leq 0 \Rightarrow x - MM(13) = 0 \Rightarrow x = MM(13)$ which means that $\forall x \neq MM(13) \Rightarrow x \notin C(13)$

   Also the first property proves that $MM(13) \in C(13)$

   By the above statements being true, the property is proven.

3. $x \in C(n) \Rightarrow |x - MM(13)| \leq (13 - n) \Rightarrow n - 13 \leq x - MM(13) \leq 13 - n$

   $y \in [x+m-n, x-m+n] \Rightarrow x+m-n \leq y \leq x-m+n \Rightarrow m-n \leq y-x \leq m+n$

   By adding the above, we deduce that $m - 13 \leq y - MM(13) \leq 13 - m \Rightarrow |y - MM(13)| \leq (13 - m)$

   Also, $(x - y) \equiv (m - n) \mod 2$ and $(x - MM(13)) \equiv (13 - n) \mod 2$ can both hold true when $(y - MM(13)) \equiv (13 - y) \mod 2$ holds true.

   So both conditions of the consistent score set definition hold true $\forall y \in [x + m - n, x - m + n]$.

4. Initially we will prove that: If $0 \leq m \leq 12$ and $y \in C(m)$ then $y+1 \in C(m+1)$ or $y - 1 \in C(m + 1)$ or both.

   $y \in C(m) \Rightarrow (y-MM(13)) \equiv (13-m) \mod 2 \Rightarrow (y+1-MM(13)) \equiv (13-m-1) \mod 2$ and $(y - 1 - MM(13)) \equiv (13 - m - 1) \mod 2$

   $y \in C(m) \Rightarrow |y - MM(13)| \leq (13 - m) \Rightarrow |MM(13) - y| \leq 13 - m \Rightarrow m - 13 \leq MM(13) - y \leq 13 - m$

   If $y > MM(13)$:

   $y > MM(13) \Rightarrow MM(13)-y+1 \leq 0 \leq 12-m \Rightarrow MM(13)-(y-1) \leq 13-(m+1)$ also $m - 13 \leq MM(13) - y \Rightarrow (m + 1) - 13 \leq MM(13) - (y - 1)$

   The above prove that $y - 1 \in C(m + 1)$

   In a similar way we can prove that $y + 1 \in C(m + 1)$ if $y < MM(13)$

   If $y = MM(13)$:

   In that case $m \neq 12$ since it has to satisfy the $(y - MM(13)) \equiv (13 - m) \mod 2$ reastriction, but if $m = 12$ it cannot. This means that $m \leq 11 \Rightarrow m-11 \leq 0 \Rightarrow m-11 \leq MM(13)-y \Rightarrow m-12 \leq MM(13)-y-1 \Rightarrow (m+1)-13 \leq MM(13)-(y+1)$.

   Also $MM(13)-y \leq 13-m \Rightarrow MM(13)-y-1 \leq 13-m-1 \Rightarrow MM(13)-(y+1) \leq 13 - (m + 1)$

   The above prove that $y + 1 \in C(m + 1)$

   We have proven that the initial statement holds true for every y and MM(13).

   Now we are going to prove that $\forall n, 0 \leq n \leq 13$ and $\forall k, 0 \leq k \leq 13 - n$ if $x \in C(n)$ then $\exists y \in [x - k, x + k] \Rightarrow y \in C(n + k)$

   If $k = 0$ the statement is true and $y = x$.

   Asuming that the statement holds true $\forall k < 13 - x \Rightarrow \exists y \in [x - k, x + k] \Rightarrow y \in C(n + k)$

   We have proven above that $y + 1 \in C(n + k + 1)$ or $y - 1 \in C(n + k + 1)$. Since $y \in [x - k, x + k]$, $y + 1 \in [x - k + 1, x + k + 1] \subseteq [x - k - 1, x + k + 1]$ and $y - 1 \in [x - k - 1, x + k - 1] \subseteq [x - k - 1, x + k + 1]$

   This proves that $\exists z \in [x - k - 1, x + k - 1] \Rightarrow z \in C(n + k + 1)$

   By means of mathematical induction, the statement is proven.

   If we substitute $k = m - n$ in that statement, we have $\forall n, 0 \leq n \leq 13$ and $\forall m, 0 \leq m - n \leq 13 - n$ if $x \in C(n)$ then $\exists y \in [x+n-m, x+m-n] \Rightarrow y \in C(m)$.

   This is the statement we wanted to prove.

   □

The following theorem is used to lift the restriction preventing the use of results of deeper searches in the search algorithm.

Theorem 4. *If* $\exists x \geq l \Rightarrow x \in C(n)$ *then* $\exists y \geq l + n - m \Rightarrow y \in C(m)$
If $\exists x \leq u \Rightarrow x \in C(n)$ *then* $\exists y \leq u - n + m \Rightarrow y \in C(m)$

Proof. We will prove the first statement.

- If $n \geq m$:

  Assuming $x \geq l \in C(n)$ then according to the third property $y = x+n-m \in C(m)$.

  $x \geq l \Rightarrow x+n-m \geq l+n-m \Rightarrow y \geq l+n-m$. This means that $\exists y \geq l+n-m \Rightarrow y \in C(m)$

- If $m > n$:

  Assuming $x \geq l \in C(n)$ then according to the fourth property $\exists y \in [x-m+n, x+m-n] \Rightarrow y \in C(m)$. This means that $\exists y \geq x-m+n \geq l-m+n \Rightarrow y \in C(m)$.

This proves the first statement. the second statement has a similar proof. □

Algorithm 1 can now be used with a minor alteration. minimax(a, b, N) is no longer required to return $MM(N)$, but returns a value $x \in C(N)$. This way it can use transposition table entries of deeper searches. This was impossible with the algorithm returning $MM(N)$. We can also use heuristics that count tricks potentially deeper than the depth being searched.

4.5 Forward pruning by using admissible heuristics

It is possible to know a lower bound of tricks won by a player without performing a complete search. Some heuristics calculate these tricks. These heuristics are considered admissible if they never overestimate the tricks. If all heuristics are admissible, the search algorithm will return an accurate result. We are now going to describe the heuristics used, all of which are admissible.

4.5.1 Quick tricks

Quick tricks are the tricks that can be won, by the player whose turn is to play, before losing a trick. It is an admissible heuristic, since tricks won that way cannot be more than the total tricks won by the current player. Our implementation though does not compute the exact number of quick tricks, because its high computational complexity gives us more pain than gain. Instead we compute an approximate value, that is never higher than the true number of quick tricks. This makes our Implementation also admissible. This heuristic works for both suit and NT stains.

We are now going to explain our Implementation. We have precomputed the number of quick tricks per suit, as well as the potential quick tricks if the partner can win a trick and weather the partner can win a trick in that suit(which is called entry). When we compute the heuristic, we add the quick tricks in each suit and the potential quick tricks if we have at least one entry. When the stain is a suit, the quick tricks in the trump suit are first determined and if they are less than the trumps of an opponent hand the quick tricks in all other suits are capped to the length of the suit in that hand.

4.5.2 Later tricks

Later tricks are the quick tricks of the opponent. However, they cannot be more than the remaining tricks minus the opponent's quick tricks. In order to make this heuristic admissible, he have to determine an upper bound of the quick tricks. Since there are two possible opponent hands, we also have to compute the quick tricks for each hand leading the trick and accept the minimum. Due to the difficulty of determining the upper bound of quick tricks when trumps are in play, we only use this heuristic for NT stains or when all trumps have been played.

4.6 Move ordering

It has been proven that the performance of the $\alpha - \beta$ pruning algorithm is heavily influeneced by the order in which the nodes of the search tree are expanded[11]. Searching the best move first is optimal, since it reduces the number of nodes expanded. However, the best move cannot be easily selected.

For our double-dummy solver, we used both problem specific heuristics and more generic ones, in order to assign a score for each move. For the problem specific ones, we used some guidelines of bridge. For example, in a no-trump contract, leading a trick from suit having a king and a low card and the right-hand opponent having the ace and the queen is considered a bad move. On the other hand, leading the same suit from partners hand can be a good move. These guidelines depend on the existence of trumps, the number of card each player has in each suit, the loccation of high cards and the number of cards that have already been played in the trick. Our implementation shares many principles with the implementation of the DDS library by Bo Haglund and Soren Hein, the documebtation of which was the main source of the guidelines[13].

The more generic guidelines is the killer heuristic (the first application of which in bridge was in GIB[12]) and hash moves. The former increases the score of moves that have caused a cutoff in siblin nodes, while the later ensures that the stored winning move is examined first in case of a hit in the transposition table.

Chapter 5

DEALING WITH IMPERFECT INFORMATION

5.1 The imperfect information in single-dummy bridge

In double-dummy bridge we have perfect information. On the other hand, in real bridge we have far from perfect information. In reality when the second card is about to be played, the declarer has to consider $\binom{25}{12}$ (the number of ways 25 cards can be dealt between two hands, the one receiving 12 cards and the other 13) possible deals. We have to play as well as possible without knowing which deal is the true one.

In the following sections we are going to develop methods that deal with the lack of information, more or less successfully. Those methods use the double-dummy solver to solve several double-dummy problems that can be possible deals of the single-dummy problem and use those solutions to solve the single-dummy problem.

5.2 The importance of the double-dummy problem in single-dummy bridge

The most important difference between double-dummy and single-dummy bridge is the fact that some cards are hidden from the player. The most common and effective way of playing single-dummy bridge is seeing the board as a number of double-dummy problems, any of which having some odds of beeing the true deal. That way single-dummy bridge can be seen as a large set of double-dummy problems. Most human bridge players do not play strictly as above. Instead, they examine thinks like the location of a high card or the number of trumps each opponent holds. Each such assumption includes many double-dummy problems that are treated as equal (usually without any negative effects on the playing line). On the other hand it is hard for a computer to make such generalisations. Most comuter bridge players, including the one implemented for this thesis, randomly generate a number of double-dummy problems hoping that every case a human would take into consideration is represented by some of the generated double-dummy problems. We are now going to provide some definitions of terms used in the rest of this thesis.

Definition 7. Consistency A double-dummy problem is consistent to a single-dummy if it can possibly be the actual deal.

World A double-dummy problem consistent to the single-dummy problem.

Sample A randomly generated world.

Weight A number representing the likelihood of a sample beeing the correct one.

World (or sample) space The set of all worlds of the single-dummy problem.

A double-dummy problem is consistent to single dummy problem if the the following restrictions are met:

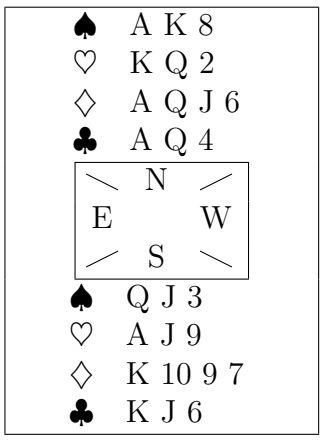
- The hands of dummy and the player to play are the same.
- The hands played before reaching the current state are legal in the double-dummy problem.
- The generated hands do not contradict to the bidding.
- The cards played by the other players are reasonable to their view of the double-dummy problem.

5.3 The sample space and information

It is clear that if the sample space has a size of 1, the single-dummy problem collapses to the perfect information double-dummy problem. On the other hand, if the sample space is large, there is a lot of missing information. In general the size of the sample space, is a metric of missing information.

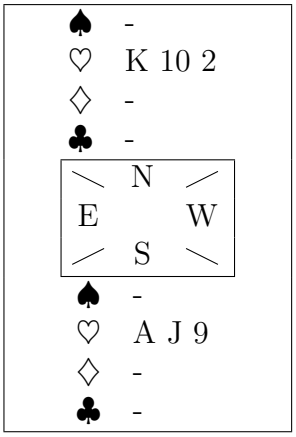
However, the amount of missinig information is not the only thing that matters. Information can be more or less valuable. Below we are going to provide two extreme examples that confirm this fact.

In the first example we have the following deal, on a 7NTS contract.



After the opening lead, the sample space will have a size of $\binom{25}{12}$. The amount of missing information is the highest possible. However the missing information is completely valueless, since there is a fullproof way that guarantees all 13 tricks for the declarer.

On the other hand, in the following position, a much lower amount of information is much more valuable:



South has to play and the remaining cards at the defenders' hands are Q♡, 8♡, 6♡, 5♡, 4♡ and 3♡. The sample space has a size of only $\binom{6}{3} = 20$, but the declarer can take all 3 remaining tricks with a propability of 50% (when the location of Q♡ is correctly guessed) and two tricks with a propability of 50%. This little missing information will cost the declarer a (potentially critical) trick 50% of the time.

5.4 The Monte-Carlo approach

This is a simplistic method to solve imperfect information problems. It is easily implemented, and it is one of the least computationally demanding methods. However it has some serious drawbacks, that cause it to give bad solutions sometimes.

We are now going to explain the algorithm.

Algorithm 2. The Monte-Carlo algorithm

1. We generate N double-dummy problems, that are possible dealings of our single-dummy problem. From now on we will call these samples.
2. We can optionally assign a "weight" to each sample that shows the likelihood of having that sample as the real one. This takes moves already played into account. Finding the exact weight is impossible, but it can be approximated. The weight of the i-st sampled will be notated as W_i.
3. We solve each double-dummy sample and get the minimax score after playing each possible move. The score is not necessarily the score used in the double-dummy solver, but it is always an increasing function of that score. The score of the i-st sample and the j-st move is notated as S_{ij}
4. Find the j that maximizes $\sum_{i=1}^{N} W_i S_{ij}$. Return the move that matches to that j.

The drawbacks of this algorithm are:

1. There is always the possibility that we generate samples that do not represent our single-dummy problem. For example most samples can be rare dealing. The chance of this happening diminishes as N increases.
2. Calculating a good approximation of the weight of a sample is not always easy. This is not a problem of the Monte-Carlo algorithm, but it negatively affects the move returned.

3. It considers that it will have perfect information after playing the current move. Information gathering is not in the goals of this method. This is the greatest drawback of the Monte-Carlo approach. This problem is widely defined as strategy fusion.

5.5 The method that we implemented

The goal of this method is to overcome the third disadvantage of the Monte-Carlo method. Initially we generate N random samples, just like the Monte Carlo method. However we do not solve them as independent double dummy problems. Instead we do the following, in order to calculate the expected payoff:

Algorithm 3. Calculation of the expected payoff

```
    payoff(state, samples)
if state = finished then
    return score(state)*sumOfWeights(samples)
else
    if the defender has to make a move then
        for each move do
            samples[move] = ∅
        end
        for each sample ∈ samples do
            for each move ∈ validMoves(state, sample) do
                Sample s
            s.world = sample.world
            s.weight = assignw(state, sample, move)
            samples[move] = samples[move] ∪ s
            end
        end
        sum = 0
    for each move do
            if samples[move] ≠ ∅ then
                doMove(state, move)
            sum = sum + payoff(st, samples[move])
            undoMove(state, move)
            end
        end
        return sum
    else
        max = −∞ for each valid move do
            doMove(state, move)
        x = payoff(st, samples[move])
        if x > max then
            max = x
        end
        undoMove(state, move)
        end
        return max
    end
end
```

When this algorithm is used to select a move for the declarer, the move that maximizes the payoff is selected.

5.5.1 Strategy and its importance

Every turn based game (including bridge) is based on responding to other player's moves. In perfect information games, like chess and double-dummy bridge, the past moves are important only because they compose the current position. In imperfect information games, on the other hand, past moves provide a source of information, in addition to composing the current position.

In general a strategy can be viewed as a tree, with two types of nodes, the ones in which our player has to play and the ones that any other player has to play. The former nodes have exactly one child if the strategy is complete (unless we have reached the end of the game), or at most one if the strategy is incomplete. The latter nodes on the other hand have one child for each possible valid move for the player whose turn is to play.

5.5.2 Intuitions

The Monte-Carlo method is good at finding a good move, but that move might not help finding good moves after that. On the other hand, we need to find a move that reduces the chance of errors (as much as possible) in the current, as well as the next moves. We also need to increase the chance th opponent makes an error as much as possible, especially if we play as defenders.

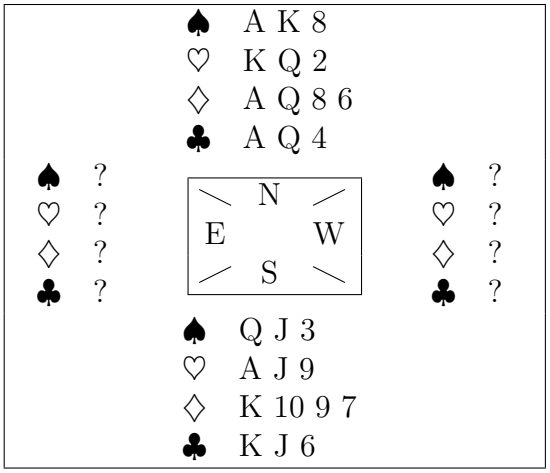
In order to achieve the above, the move we make has to be part of a strategy. This is what the Monte-Carlo method fails to achieve, and what our method tries to achieve. What our method does is think about "what am I going to play next?". For each sequence of responces of the other players, only one move may be considered.

The strategy generated by our method tries to achieve two things more effectively than the Monte-Carlo method. These are:

- Obtain the information needed to make a future guess more likely to succeed.
- Play in such a way that eliminates the need to guess at the future.

Below are two typical examples in bridge, where the Monte-Carlo aproach may fail to find the optimal strategy, while our method succeeds.

The first example is the following hand, on a 7NT contract. Suppose that the declarer is South. Suppose that the oppening lead is 5♠(it could be anything but a diamond).



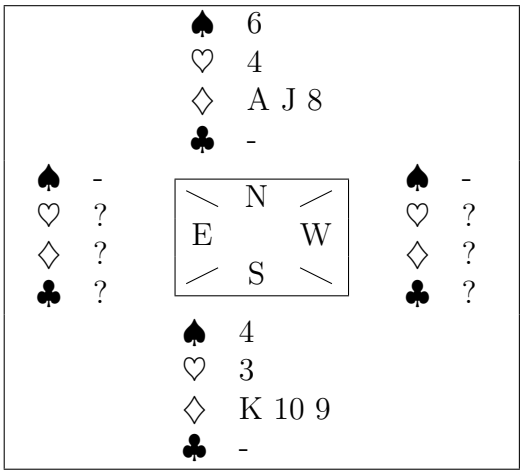
The above contract can easily succeed if the remaining diamonds break 2-3 (which means East or West holds 3 diamonds and the other defender the remaining two) or the

J♢ is singleton. If this is not the case however, the contract will fail if the location of the J♢ is not guessed correctly. There is no guarantee that the key information can be gathered, but there is a way to maximize the odds of succeeding in the contract.

A way to maximize the odds is to play the A♢ (or the Q♢) as soon as possible. If the J♢ drops, there is no way to fail. If one defender discards (disclosing the location of all diamonds), the contract can succeed by taking the proven fineese. In any other case, we play the rest of the diamonds last. The information gathered from the oponents' discards may give us the information needed to secure the contract (or at least increase the odds of a correct guess).

The Monte-Carlo approach will detect an illusion of 13 certain tricks. The moves played until this deception is uncovered may mean the difference between success and failure.

The second example is the "Ruff and sluff endplay", as shown below. Suppose that it is the turn of South (declarer) to lead the ninth trick, and that the trump suit is ♠.



The remaining cards at the hands of the defenders are: 8♡, 7♡, 6♡, Q♢, 6♢, 5♢, 4♢, 3♢, 2♢ and 5♣. However the location of these cards has not been disclosed to the declarer. The only way that guarantees 4 tricks for the declarer is 3♡. Every other line of play relies on the correct guess of the location of the Q♢, in order to achieve 4 tricks. While our method will play correctly, the Monte-Carlo method will incorrectly "think" that every line of play, with the exeption of K♢ guarantees 4 tricks. The correct line of play, does not give any valuable information to the declarer. Instead, it renders a piece of information (the location of the Q♢) valueless. Instead of increasing the chances of a successful guess at the location of the Q♢, it eliminates the need to guess.

5.5.3 Optimizing this method for declarer play

Single dummy bridge is played by three players at the playing phase. The declarer (who also playes dummys hand) and the two defenders (who play only their own hand). All three players know their own hand and the dummys hand. The fact that the side of the declarer usually makes more bidds than the defenders in the auction, allows us to assume that the defenders can obtain more information from the bidding phase about the declarer's hand than the opposite. Also the fact that the defenders are teammates, allows them to exchange signals about their hand to each other. This gives the defenders another information advantage. The above usually makes imperfect information a crucial

problem for the declarer and rarely a problem for the defenders. This makes most human declarers consider the oponent's defence perfect. The propability that this assumption leads to suboptimal declarer play is negligible. The opening lead is the only exception to the previous rule, however this move is only played before the first move of the declarer and can be excluded from the perfect defence assumptions.

Our agent, when playing as declarer can also use this assumption, making our search algorithm less complex. This assumption allows us to assign non-zero propabilities (Pr_{jik}) to only optimal future defender moves. The opening lead does not affect the assumption since it is a past move.

Theorem 5. *The expected payoff of each declarer move estimated by our method under the perfect defence assumption cannot be greater than the expected payoff estimated by the monte-carlo aproach.*

Proof. The payoff of estimated by the Mont-Carlo method equals the weighted sum (or average) of the payoffs of each double dummy sample when both sides play optimally (the minimax value of the sample after the move examined has been played). On the other hand, the expected payoff of a move estimated by our method equals the weighted sum (or average) of the payoffs of each double dummy sample asuming the defenders optimally (the declarer might or might not play optimally). When the defenders play optimally, no payoff in a double-dummy sample can be greater than its minimax value, so the payoff of each sample estimated by our method cannot be greater than that of the Monte-Carlo method. This means that the weighted sum(or average) of the payoffs estimated by our method cannot be greater than the sum of those estimated by the Monte-Carlo method. □

The above theorem allows us to use a prunning technique. Before aplying our algorithm, we can determine the monte-carlo payoff of each move, order the moves by descending monte carlo payoff and start aplying our algorithm to the moves with the higher payoffs. When the moves that have not been tested by our algorithm have a lower payoff than the max payoff of the moves tested with our method, we can abort the search and return the move with the highest payoff. This does not change the result of our algorithm.

5.5.4 Spot card relocation

In bridge, cards tend to be distinguished between high cards and spot cards (or low cards). High cards are the cards with a rank high enough to win a trick, when other cards of the same suit are played in the coresponding trick. Spot cards on the other hand are not expected to win such tricks. In general, human players tend to treat each spot cards as x, which means any low card. Humans can intuitively judge if the rank of a card matters (and treat it as a high card) or not (and treat it as x). Humans tend to plan their play in ways like "If both opponents follow suit, do something, else do something else". In the previous example, only the length of the suit of each opponent is important and the rank of each card irrelevant. In that case each card in the opponents hand (in the specific suit) can be treated as x.

The above method can be useful to a computer bridge player. Treating low cards as x and treating each x responce by the oponents as equal (and thus not distinguish two or more worlds that respond with a different spot card), can depen the search of our

algorithm without increasing the number of samples. This will allow as to find deeper strategies, while keeping the number of samples relatively low. In order to simulate that, and thus make the algorithm more effecttive, we can randomly order all the spot cards that are in the defender's hands and give the top ones to one and the bottom ones to the other. This however might give incorrect result if we treat a high card as a spot card.

Finding a way for a computer to distinguish high cards and spot cards is not easy. An aproximate method, which is used in this thesis is treating half of the outstanding cards (the ones with the lowest rank) in each suit as spot cards.

The algortihm used for spot card relocation is the following:

Algorithm 4. spot card relocation

```
    relocation(mcHands[samples])
for each suit do
    |   spots = array of the spot cards in the defender's hands
    |   randomly shuffle spots
    |   for i = 0; i < samples; i++ do
    |   |   l1 = number of spot cards in the hand of one defender at sample i and the current suit
    |   |   l2 = number of spot cards in the the other defender's hand at sample i and the current suit
    |   |   the spot cards of the first deffender are swapped with l1 top cards of the spots array
    |   |   the spot cards of the other deffender are swapped with l2 bottom cards of the spots array
    |   |   end
    |   end
```

5.6 Aproximating the sample weight

Both the Monte-Carlo aproach and our implementation assign a weight (W_i) to each sample. The weight should be proportional to the propability this sample is the correct world, considering the moves of the other players. If all valid samples have equal chances of beeing selected, the weight can be the propability the players play the moves played considering the sample is the correct world. Since effectively finding the exact weight is impossible, we need to find an aproximate value. For our research, we have tested the effectiveness of three different aproaches.

1. Completely ignore weights. Each sample is assigned a weight value of 1.

2. Assume a random play by the other players. Each sample is initially assigned a weight of 1 and each turn from the begining until the current state is devided by the number of valid moves.

3. Assume a propability of correct play P and judge each sample by the correctness of each move. This aproach looks like the previous one but the weight of correct moves each turn is multiplied by P and devided by the number of correct moves, while the weight of incorrect moves is muliplied by $1 - P$ and devided by the number of incorrect moves. This requires an anylysis of the past moves, which greatly adds to the complexity of the problem. Completely adopting the perfect defence assumption will set $P = 1$. Using simple machine learning techniques can lead us to an aproximation of P dependent on the current turn, the number of correct and incorrect moves and the player who has to play.

The third method is by far the most consistent, and by far the most complex.

5.7 Defender play

Our declarer play is based on the perfect defence assumption. This basically means that the declarer will play as if the the defenders play doubly-dummy bridge. However the defender cannot assume that the declarer will play perfectly. Also in real world bridge the defender has imperfect information as well, although the lack of information is more crucial for the declarer. In real world bridge, the defenders play in such a way that disclose information about their hand to their teamate (the declarer inevitably gets this information as well), through signaling conventions. However most (if not all) bridge playing agents ignore signaling. The defender also tries to exploit the declarer's lack of information, in order to prevent him from playing optimally.

5.7.1 Optimizing perfect defence

Any defensive play that does not lose tricks in the double-dummy deal is considered perfect. However, this does not guarante the most tricks for the defender since it does not necessarily exploit the declarer's lack of information. As a result, the declarer can make fewer errors.

An optimized perfect defender was implemented, in order to increase the possibility of an error by the declarer. A small number of samples (from the declarer's point of view), that are used in order to find an expected response by the declarer (based on either the Monte-Carlo method, or our method). Among the good moves for the defender, the one that expects the most errorous response is selected.

Part III

EXPERIMENTS, CONCLUSIONS AND FUTURE WORK

Chapter 6

TESTING OUR AGENT

6.1 The testing procedure

For each test set we have generated 100 different random complete deals that satisfy some restrictions. For each defender play, we select either one of the correct moves or the move selected by the optimized defender, while the declarer plays according to the methods described above. We use both the Monte-Carlo and our implementation (both with spot card relocation and without) for the third weighting aproach with $P = 1$. We test each number of samples between 1 and 100 for each declarer move. The scoring scheme is the score the declarer will get for the specific number of tricks.

The test sets are:

1. 1NT and 1♡ makeable partscore cantracts.
2. 3NT, 4♡ and 5♢ makeable game cantracts.
3. 6NT and 6♡ makeable slam cantracts.
4. 7NT and 7♡ makeable grand slam cantracts.

Finally we are going to give an aggregation of the above tests, by giving each test set a weight representing the the frequency of similar contracts apearing in Bridge tournaments. This will give us an estimation of the usefulness of the methods in real world situations.

The perfect score minus the actual score can be considered as value of information, since it is 0 for perfect information and gets lower as we deal with the lack of information more effectively. Comparing the results of each method can show us which is better, in the specific test set. We also show the effect of increasing the number samples.

The experiments are oriented towards the following areas:

1. Comparison between the Monte Carlo method and our method against perfect defence that selects an arbitrary perfect move.
2. Comparison of our method with and without spot card relocation against perfect defence that selects an arbitrary perfect move.
3. Comparison of the above methods against the optimized perfect defender.
4. Comparison of the arbitrary perfect defender and the optimized perfect defender against the three declarer aproaches.
5. Comparison of the execution times of the tested declarer methods.

6.2 The performance of our method compared to the Monte - Carlo aproach

Below are the aggregated results of the experiments:

Table 6.1: Comparison between the Monte-Carlo method and our method

No of	Monte Carlo				Our Method				IMP
Samples	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	Difference
1	31.01	26.23	42.76	176.63	27.49	28.65	43.86	165.62	-0.16
2	32.86	25.54	41.61	190.78	31.80	25.40	42.80	187.13	-0.12
3	38.18	26.82	34.99	240.54	37.28	25.72	37.00	229.32	-0.19
4	40.00	26.05	33.95	248.60	40.20	24.36	35.44	250.44	0.03
5	43.07	23.57	33.36	258.65	39.86	25.98	34.16	259.37	-0.04
6	46.10	24.15	29.74	287.64	46.20	24.60	29.20	301.26	0.19
7	47.30	25.59	27.11	301.42	50.68	21.75	27.58	300.92	-0.01
8	48.70	24.56	26.75	312.85	49.02	22.46	28.52	297.95	-0.24
9	50.05	22.71	27.24	307.39	49.47	23.66	26.87	310.13	0.06
10	47.14	23.31	29.55	285.56	54.89	22.22	22.89	331.37	0.74
12	52.89	20.09	27.01	309.45	56.50	19.89	23.61	330.41	0.33
14	53.41	22.34	24.25	327.79	56.80	22.27	20.93	348.28	0.34
16	55.54	21.86	22.60	331.52	59.91	19.34	20.76	357.02	0.37
18	56.22	20.85	22.93	338.50	59.39	19.11	21.50	355.85	0.27
20	58.69	20.60	20.71	356.69	57.94	19.35	22.71	344.94	-0.19
23	60.13	20.92	18.95	370.49	63.15	18.51	18.34	374.53	0.05
26	63.37	16.77	19.86	366.93	63.59	18.64	17.77	382.60	0.25
30	57.79	20.31	21.90	351.57	64.76	18.38	16.86	382.65	0.48
35	66.30	18.43	15.27	400.55	69.02	16.31	14.67	402.17	0.07
40	63.74	17.51	18.75	372.03	66.47	17.18	16.35	398.04	0.38
50	66.86	17.93	15.21	402.96	65.41	18.13	16.45	388.18	-0.22
60	66.37	17.30	16.33	390.16	69.67	16.09	14.24	401.76	0.19
80	68.08	16.75	15.17	402.36	70.93	16.64	12.43	416.05	0.25
100	68.43	17.71	13.86	414.13	71.76	16.15	12.09	418.82	0.11

When the number of samples is low (fewer than 10), both methods have similar performance. Sometimes the Monte - Carlo aproach is better, while other times our method is better. This is expected since our method gives the exact same results with the Monte - Carlo method when the number of samples is low. The fact that the number of samples is low is the reason for bad play from both declarers as well as the main reason for flancuations in scores.

While the number of samples increases, our method starts gaining a slight advantage over the Monte - Carlo method. At 100 samples our method gains 0.11 IMPs/board in the aggregated results, however at 80 samples we have 0.25 IMPs/board. 0.11 IMPs/board is a slight but significant edge in high level bridge tournaments. The fact that the most dicisive part of bridge is the bidding, makes this slight edge important.

In suit contracts both methods seem to perform almost equally (tables A.1, A.4, A.5, A.6, A.8), while our method is far superior in no-trump contracts (tables A.2, A.3, A.7, A.9). This might have two possible explanations. Either the Monte-Carlo method is good enough for suit contracts, or the number of samples are too few to make a difference. In either case, our method seems to work pretty well in NT contracts, while the gain is minimal in suit contracts.

The level of the contract plays the expected role in the results. The higher the level the higher the gain. This is expected, since high level contracts punish mistakes

harder. However the highest gain is in slam contracts (tables A.6, A.7), not in grand slams (tables A.8, A.9). This can be explained by the fact that some strategies that our method employs require sacrificing a trick at the right moment, which is out of the question in grand slams.

6.3 The effects of the spot card relocation improvement

Below are the aggregated results of the experiments:

Table 6.2: Comparison between our method without spot card relocation and with relocation

No of	No Relocation				Relocation				IMP
Samples	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	Difference
1	27.49	28.65	43.86	165.62	30.47	26.40	43.13	170.29	0.06
2	31.80	25.40	42.80	187.13	32.47	29.44	38.10	215.30	0.51
3	37.28	25.72	37.00	229.32	37.00	24.81	38.18	224.35	-0.10
4	40.20	24.36	35.44	250.44	37.15	26.58	36.27	236.87	-0.25
5	39.86	25.98	34.16	259.37	42.86	26.42	30.72	273.62	0.33
6	46.20	24.60	29.20	301.26	46.77	22.68	30.54	278.92	-0.34
7	50.68	21.75	27.58	300.92	49.41	21.87	28.72	293.86	-0.13
8	49.02	22.46	28.52	297.95	51.92	22.40	25.69	323.26	0.39
9	49.47	23.66	26.87	310.13	53.77	21.90	24.33	330.61	0.35
10	54.89	22.22	22.89	331.37	55.51	21.43	23.06	340.22	0.11
12	56.50	19.89	23.61	330.41	56.74	20.59	22.67	345.13	0.21
14	56.80	22.27	20.93	348.28	59.81	19.40	20.79	360.71	0.22
16	59.91	19.34	20.76	357.02	60.93	19.86	19.21	363.46	0.11
18	59.39	19.11	21.50	355.85	62.41	18.24	19.35	359.69	0.12
20	57.94	19.35	22.71	344.94	63.85	18.45	17.70	372.26	0.49
23	63.15	18.51	18.34	374.53	64.17	19.16	16.66	388.50	0.22
26	63.59	18.64	17.77	382.60	66.99	17.30	15.72	391.90	0.17
30	64.76	18.38	16.86	382.65	65.75	17.27	16.98	391.08	0.15
35	69.02	16.31	14.67	402.17	67.36	17.50	15.14	400.03	-0.08
40	66.47	17.18	16.35	398.04	70.84	14.44	14.72	410.00	0.23
50	65.41	18.13	16.45	388.18	72.94	14.04	13.02	415.75	0.50
60	69.67	16.09	14.24	401.76	73.74	13.66	12.60	423.00	0.31
80	70.93	16.64	12.43	416.05	77.32	11.40	11.27	429.54	0.23
100	71.76	16.15	12.09	418.82	75.60	13.91	10.49	430.94	0.21

With extremely low number of samples, spot card relocation has no effect. In that case, all three methods perform about the same. However, as the number of samples increasess, spot card relocation starts gaining a good IMPs/board. The highest number on the aggregated results is 0.5 at 50 samples, that diminishes to 0.22 at 100 samples. This is a tremendous gain, considering that the previous Bermuda Bowl finals ended in a difference of 15.5 IMPs over 128 boards (and most of that difference was due to different bidding).

Suit contracts gain more from spot card relocation (tables A.11, A.14, A.15, A.16, A.18). In no-trump contracts (tables A.12, A.13, A.17, A.19) spot card relocation might lose with 100 samples and win marginally with fewer. This means that the drawbacks of relocation tend to outweigh its advantages unless the number of samples is relatively low. In suit contracts however, relocation dominates. This leads us to the conclusion

that in suit contracts deeper strategy is required than in NT contracts. It also leads us to the conclusion that our method with 100 or less samples without relocation kind of fails because of the deeper strategy is required.

The level of the contract has the expected effect. The higher the contract, the bigger the gain or the loss.

6.4 The performance against the optimized defenders

Below are the aggregated results of the experiments:

Table 6.3: Comparison between the Monte-Carlo method and our method

No of	Monte Carlo				Our Method				IMP
Samples	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	Difference
1	20.03	28.19	51.79	93.35	32.07	25.45	42.48	173.53	1.39
5	42.87	23.34	33.79	250.00	45.03	22.99	31.98	274.97	0.39
10	50.45	21.11	28.44	303.04	53.47	20.92	25.62	313.84	0.24
20	51.38	24.05	24.57	337.71	61.36	20.13	18.51	371.14	0.62
35	63.00	19.73	17.26	377.60	67.43	17.83	14.74	398.20	0.31
50	65.67	17.11	17.22	382.43	66.73	18.30	14.97	395.90	0.28
80	65.20	17.70	17.11	381.52	70.27	16.89	12.83	414.33	0.57
100	63.24	19.37	17.39	388.73	70.05	15.63	14.32	399.28	0.23

Table 6.4: Comparison between our method without spot card relocation and with relocation

No of	No Relocation				Relocation				IMP
Samples	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	Difference
1	32.07	25.45	42.48	173.53	29.58	27.02	43.39	168.55	-0.02
5	45.03	22.99	31.98	274.97	47.01	22.25	30.74	266.68	-0.02
10	53.47	20.92	25.62	313.84	55.69	22.01	22.30	345.69	0.50
20	61.36	20.13	18.51	371.14	64.27	17.53	18.21	370.88	0.01
35	67.43	17.83	14.74	398.20	65.47	18.51	16.02	389.79	-0.15
50	66.73	18.30	14.97	395.90	69.06	16.92	14.02	406.70	0.17
80	70.27	16.89	12.83	414.33	69.71	15.63	14.67	400.29	-0.19
100	70.05	15.63	14.32	399.28	69.03	15.80	15.17	393.88	-0.12

Against the optimized defender, whith 100 samples our method gains on the Monte-Carlo aproach the significant score of 0.23 IMPs/board, which is way higher than the one against the arbitrary defender. The benefit is greater in low level suit contracts, while it becomes insignificant in higher level and No-trump contracts. 3NT (table B.3), 5♢ (table B.5)and 6♡ (table B.6) give a negative score for our method, while 1♡ (table B.1) gives a score of 0.54 IMPs/borad. With fewer samples the gain is even bigger for our method especially for higher level contracts.

The spot card rellocation has an unexpected detrimental effect on the score, losing on average 0.12 IMPs/board against the method without rellocation with 100 samples (table B.20). With a smaller number of samples (for instance 50), spot card relocation seems to have a small negative or positive effect. The stain and the level of the contract, does not seem to affect the outcome significantly.

6.5 Comparing the two defenders

Below are the aggregated results of the experiments:

Table 6.5: Aggregated Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	174.84	93.35	1.40	163.69	173.53	-0.13	168.62	168.55	-0.01
5	255.43	250.00	0.08	256.14	274.97	-0.34	270.87	266.68	0.03
10	280.95	303.04	-0.24	328.02	313.84	0.30	335.93	345.69	-0.11
20	352.69	337.71	0.34	340.37	371.14	-0.48	367.74	370.88	0.00
35	396.32	377.60	0.28	398.11	398.20	0.05	395.68	389.79	0.12
50	398.74	382.43	0.31	383.18	395.90	-0.20	411.60	406.70	0.14
80	397.84	381.52	0.33	411.43	414.33	0.00	424.80	400.29	0.43
100	410.05	388.73	0.41	414.20	399.28	0.28	426.75	393.88	0.60

The optimized defender seems to perform better against any declarer. At 100 samples, the optimized defender gains 0.41 IMPs/board against the Monte-Carlo declarer, 0.28 against our implentation of the declarer and 0.6 aginst the declarer that uses spot card relocation. At a lower number of samples, the gain is deminishes. The defender is more sucessful against No trump contracts (tables C.2, C.3, C.7, C.9), while it fails misserably against 7♡ contracts (table C.8).

6.6 The effect of each method on execution times

Below are the aggregated results of the experiments:

Table 6.6: Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	4.2	4.3	4.3
5	19.2	23.3	24.5
10	35.2	49.3	55.0
20	64.9	116.1	145.0
35	105.9	231.4	299.4
50	144.3	386.7	521.7
80	225.7	780.7	1166.1
100	278.1	1139.9	1663.8

Whith one sample, every method takes the exact same time, as expected.

In the Monte-Carlo approach, the execution time increases linearly with the number of samples, as expected. The execution time of our method however, increases much faster. When spot card relocation is used, even more time is needed. This is expected, since all three method begin a Monte-Carlo execution and the relocation forces a deeper search for the same number of samples. However, the execution time hit is bigger than expected. A possible reason for that is the implementation of the transposition table of the DDS library, because it is thread specific and might be cleared too often. The execution time does not vary significantly among different contracts. The only decisive factor is the number of the samples.

Spot card relocation further increases the execution time, since it forces an even deeper search. The implentation problems of the transposition tables, cause a greater

performance hit in that case, since the generated double dummy problems are more similar, and cause more hits across different samples. In this case, the stain and the level of the contract do affect the execution time. In general, grand slam contracts (tables D.8, D.9) take the biggest performance hit, followed by slam contracts (tables D.6, D.5). Also suit contracts (tables D.1, D.4, D.5, D.6, D.8) tend to take longer time than No-trump contracts (tables A.12, A.13, A.17, A.19). A possible explanation for that is the fact that in suit contracts, deeper strategy is involved, while in high level contracts (especially 7NT) the opponents have a lot of spot cards, and spot card relocation increases the depth more than in other contracs. The fact that 1♡ (table D.1) contracts tend to take long to execute, looks like a paradox, but it can be justified by the fact that it is the most strategic.

Finally, we observe that, in the time it takes to execute the agent using the Monte-Carlo aproach with 100 samples, we can execute the agent implementing out method without relocaton with 35-50 samples and 20-35 with relocation. Against the improved defender, the Monte-Carlo agent with 100 samples scores an average of 388.73 points while our agent without relocation with 35 samples scores an average of 398.2 points.

6.7 Summarizing the results

The Monte-Carlo method, performs moderately well for declarer play. However it fails to find the optimal play in a significant number of deals. Its complexity scales linearly with the number of samples.

The method we implemented, while it generally succeeds in playing batter than the Monte-Carlo method, the cost in perfomance is high. Albeit the performance hit, it is a worthwile improvement, since reducing the number of samples, in order to meet a potential time constraint, and using our method tends to produce better moves than using the Monte-Carlo method with a higher number of samples.

On the other hand, spot card relocation, does not work well against the optimized defender, while it performs pretty well against the arbitrary perfect defender. The use of spot card relocation leads to significant increase in the execution time and is not worth using without further improvements.

In general the improved defender is far better than the arbitrary one. From that fact we can draw the conclusion that design and implementation of a realistic defender (one that does not have perfect information) that attempts to conceal information, is worthwile research.

Chapter 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusion

In this thesis we designed and implemented parts of an automated bridge playing agent. These parts include the double-dummy solver, which is used in every aspect of computer bridge and three different implementations of declarers. These are a Monte-Carlo declarer, a declarer based on a novel method and a declarer based on the same novel method using spot card relocation. We also implented a perfect information defender that selects an arbitrary perfect move and a defender that selects a perfect move that is more likely to make the declare guess.

In order to implement these parts of the agent, a significant amount of theoretical work had be caried out. The correctness of the double dummy solver had be mathematicaly proven and our method for declarer play and its optimizations have some theoretical background.

Finally, we tested our work experimentally and found out that our method plays better bridge than the Monte-Carlo method, at the cost of higher complexity. We found out that without further optimizations, the novel method that we implemented can perform better than the Monte-Carlo method, if we adjust the number of samples in a way that the execution times are equal. On the other hand, spot card relocation, although it performs better against the arbitrary defender, the complexity it adds makes it unworthy of use. Finally we found out that the improved defender can cause trouble to the declarer.

7.2 Future work

A lot of work can still be done in computer bridge. Declarer play is only a part of bridge, and it has not been exhausted in this thesis. Other subproblems of computer bridge include the bidding phase, defence play and claims.

Our method assumes that the defenders will play perfectly. In real life bridge they do not always play perfectly. A good human declarer will aim for imperfect defence, when possible. In fact such play usually does not work, but the opposite never works. Currently no automated declarer aims for a defence error. A subject of future research can be building such a declarer.

Defensive play needs to be improved as well. The defenders implemented in this thesis

cannot be used in practice, because of imperfect information. An imperfect defender that takes potential declarer errors into account, is another subject of future research.

Bidding is the part where computer bridge falls behind the most compared to humans. It is also the most decisive one, since most big score differences are caused by different biddings. Improving automated bidding whould bring computer bridge one step closer to defeating the top humans.

Finally, the methods that we implemented can be further optimized. The double dummy solver can be even faster, other performance optimizations for our method may be available and the distinction between spot cards and high cards may be more accurate.

Part IV

APPENDICES

Appendix A

RESULT TABLES AGAINST ARBITRARY PERFECT DEFENDER

Comparison of the Monte-Carlo method and our method

Table A.1: 1♡ Results

No of	Monte Carlo				Our Method				IMP
Samples	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	Difference
1	38.00	36.00	26.00	54.40	31.00	45.00	24.00	54.00	-0.04
2	30.00	43.00	27.00	52.00	29.00	42.00	29.00	49.60	-0.07
3	35.00	42.00	23.00	59.80	40.00	38.00	22.00	68.60	0.25
4	38.00	39.00	23.00	62.80	40.00	35.00	25.00	63.40	-0.01
5	40.00	38.00	22.00	65.00	35.00	37.00	28.00	47.30	-0.46
6	42.00	35.00	23.00	62.10	37.00	38.00	25.00	62.00	-0.02
7	44.00	36.00	20.00	72.70	49.00	33.00	18.00	76.10	0.11
8	42.00	37.00	21.00	71.20	44.00	34.00	22.00	68.40	-0.04
9	45.00	33.00	22.00	67.80	43.00	38.00	19.00	75.10	0.19
10	45.00	37.00	18.00	79.20	53.00	34.00	13.00	91.90	0.37
12	55.00	26.00	19.00	79.90	53.00	32.00	15.00	82.80	0.06
14	52.00	30.00	18.00	79.60	52.00	32.00	16.00	79.00	0.00
16	51.00	35.00	14.00	86.70	54.00	29.00	17.00	82.30	-0.08
18	52.00	32.00	16.00	82.50	55.00	28.00	17.00	85.60	0.12
20	51.00	33.00	16.00	79.60	57.00	27.00	16.00	85.10	0.16
23	53.00	32.00	15.00	85.30	59.00	27.00	14.00	88.70	0.09
26	62.00	22.00	16.00	88.60	57.00	27.00	16.00	85.90	-0.08
30	50.00	32.00	18.00	80.30	58.00	30.00	12.00	92.60	0.35
35	58.00	29.00	13.00	92.80	66.00	21.00	13.00	93.50	0.04
40	65.00	23.00	12.00	98.30	61.00	23.00	16.00	89.90	-0.24
50	61.00	25.00	14.00	92.50	58.00	31.00	11.00	97.10	0.11
60	63.00	24.00	13.00	95.90	71.00	20.00	9.00	103.30	0.21
80	64.00	23.00	13.00	95.50	69.00	22.00	9.00	103.90	0.24
100	60.00	25.00	15.00	90.60	68.00	22.00	10.00	99.90	0.26

Table A.2: 1NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	32.00	32.00	36.00	17.70	33.00	29.00	38.00	14.00	-0.10
2	38.00	29.00	33.00	23.40	45.00	24.00	31.00	44.80	0.47
3	49.00	28.00	23.00	50.80	43.00	21.00	36.00	18.40	-0.72
4	57.00	22.00	21.00	67.90	49.00	22.00	29.00	45.60	-0.51
5	49.00	25.00	26.00	56.70	53.00	25.00	22.00	63.50	0.20
6	58.00	23.00	19.00	72.80	54.00	24.00	22.00	64.90	-0.19
7	54.00	29.00	17.00	69.50	55.00	23.00	22.00	62.50	-0.20
8	60.00	21.00	19.00	76.20	55.00	29.00	16.00	77.60	0.04
9	60.00	27.00	13.00	85.60	59.00	24.00	17.00	82.40	-0.08
10	56.00	23.00	21.00	67.00	61.00	24.00	15.00	86.10	0.46
12	58.00	33.00	9.00	93.20	67.00	23.00	10.00	92.40	-0.01
14	61.00	25.00	14.00	87.50	69.00	18.00	13.00	92.40	0.15
16	62.00	26.00	12.00	89.10	65.00	24.00	11.00	96.60	0.22
18	63.00	25.00	12.00	92.00	68.00	18.00	14.00	86.30	-0.09
20	65.00	22.00	13.00	93.00	58.00	22.00	20.00	80.70	-0.34
23	65.00	22.00	13.00	94.20	69.00	20.00	11.00	93.50	-0.02
26	62.00	25.00	13.00	90.70	72.00	15.00	13.00	91.30	0.03
30	61.00	26.00	13.00	91.80	70.00	19.00	11.00	94.90	0.12
35	68.00	21.00	11.00	97.00	71.00	20.00	9.00	100.40	0.11
40	71.00	20.00	9.00	104.10	70.00	18.00	12.00	93.30	-0.25
50	68.00	21.00	11.00	98.00	68.00	23.00	9.00	100.20	0.06
60	72.00	20.00	8.00	106.60	75.00	16.00	9.00	101.80	-0.09
80	72.00	20.00	8.00	102.30	72.00	18.00	10.00	100.10	-0.06
100	71.00	20.00	9.00	101.70	78.00	15.00	7.00	109.20	0.22

Table A.3: 3NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	27.00	29.00	44.00	262.40	23.00	26.00	51.00	216.70	-0.74
2	37.00	23.00	40.00	306.70	34.00	24.00	42.00	295.50	-0.15
3	43.00	25.00	32.00	375.60	41.00	24.00	35.00	346.10	-0.43
4	49.00	25.00	26.00	424.40	47.00	19.00	34.00	364.70	-0.82
5	54.00	15.00	31.00	383.10	50.00	24.00	26.00	429.60	0.84
6	56.00	19.00	25.00	427.80	58.00	20.00	22.00	454.60	0.38
7	61.00	20.00	19.00	478.30	55.00	16.00	29.00	399.60	-1.36
8	56.00	20.00	24.00	441.10	57.00	16.00	27.00	419.40	-0.41
9	55.00	21.00	24.00	436.30	58.00	18.00	24.00	447.20	0.26
10	53.00	20.00	27.00	417.80	56.00	22.00	22.00	449.40	0.51
12	61.00	17.00	22.00	450.00	66.00	16.00	18.00	489.70	0.65
14	60.00	20.00	20.00	470.00	64.00	17.00	19.00	473.00	0.13
16	61.00	19.00	20.00	473.70	67.00	12.00	21.00	470.20	-0.05
18	66.00	10.00	24.00	439.50	66.00	14.00	20.00	470.60	0.52
20	66.00	15.00	19.00	484.00	63.00	19.00	18.00	488.30	0.09
23	67.00	17.00	16.00	506.60	69.00	15.00	16.00	505.70	0.02
26	66.00	15.00	19.00	481.80	75.00	13.00	12.00	534.30	0.89
30	67.00	16.00	17.00	494.90	67.00	15.00	18.00	486.10	-0.20
35	78.00	13.00	9.00	556.10	74.00	11.00	15.00	517.50	-0.66
40	67.00	14.00	19.00	486.40	69.00	13.00	18.00	494.40	0.14
50	74.00	15.00	11.00	541.80	75.00	11.00	14.00	522.00	-0.30
60	75.00	13.00	12.00	538.10	73.00	17.00	10.00	551.10	0.19
80	76.00	13.00	11.00	545.00	72.00	14.00	14.00	522.40	-0.37
100	72.00	14.00	14.00	521.10	74.00	15.00	11.00	542.10	0.34

Table A.4: 4♡ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	29.00	20.00	51.00	233.80	26.00	23.00	51.00	236.40	0.11
2	30.00	17.00	53.00	223.60	29.00	18.00	53.00	215.60	-0.24
3	36.00	21.00	43.00	305.50	30.00	25.00	45.00	285.60	-0.42
4	32.00	23.00	45.00	284.50	33.00	25.00	42.00	313.70	0.53
5	35.00	22.00	43.00	307.10	32.00	25.00	43.00	298.30	-0.21
6	39.00	24.00	37.00	353.40	43.00	22.00	35.00	369.80	0.31
7	40.00	25.00	35.00	358.00	48.00	21.00	31.00	401.00	0.80
8	44.00	24.00	32.00	389.50	45.00	21.00	34.00	370.70	-0.29
9	49.00	18.00	33.00	384.50	46.00	20.00	34.00	375.70	-0.12
10	40.00	20.00	40.00	330.50	52.00	17.00	31.00	396.50	1.06
12	41.00	19.00	40.00	340.50	50.00	15.00	35.00	374.60	0.62
14	46.00	23.00	31.00	397.40	51.00	24.00	25.00	439.50	0.72
16	54.00	16.00	30.00	411.60	57.00	21.00	22.00	468.00	0.90
18	51.00	21.00	28.00	424.90	56.00	18.00	26.00	440.40	0.22
20	57.00	18.00	25.00	449.40	51.00	18.00	31.00	403.50	-0.80
23	59.00	18.00	23.00	467.40	60.00	17.00	23.00	460.70	-0.10
26	62.00	16.00	22.00	471.50	59.00	21.00	20.00	487.00	0.29
30	56.00	16.00	28.00	426.60	65.00	15.00	20.00	483.50	0.86
35	63.00	17.00	20.00	487.90	68.00	19.00	13.00	538.60	0.82
40	56.00	20.00	24.00	456.20	67.00	19.00	14.00	531.80	1.21
50	66.00	17.00	17.00	510.50	61.00	14.00	25.00	450.60	-0.98
60	62.00	18.00	20.00	489.50	64.00	16.00	20.00	491.20	0.03
80	63.00	16.00	21.00	481.30	70.00	18.00	12.00	546.80	1.06
100	72.00	17.00	11.00	557.00	70.00	17.00	13.00	540.80	-0.25

Table A.5: 5♢ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	22.00	17.00	61.00	133.80	25.00	15.00	60.00	150.20	0.39
2	38.00	14.00	48.00	238.80	31.00	12.00	57.00	178.40	-0.92
3	30.00	16.00	54.00	198.20	32.00	12.00	56.00	190.20	-0.08
4	37.00	14.00	49.00	239.00	37.00	16.00	47.00	260.60	0.31
5	46.00	14.00	40.00	310.60	38.00	16.00	46.00	259.40	-0.95
6	46.00	16.00	38.00	319.80	48.00	14.00	38.00	325.80	0.17
7	43.00	17.00	40.00	315.20	46.00	14.00	40.00	295.80	-0.39
8	55.00	14.00	31.00	375.40	49.00	14.00	37.00	331.80	-0.72
9	45.00	19.00	36.00	339.80	48.00	15.00	37.00	325.40	-0.23
10	53.00	11.00	36.00	336.20	57.00	13.00	30.00	384.00	0.82
12	55.00	13.00	32.00	371.60	53.00	14.00	33.00	360.80	-0.20
14	55.00	13.00	32.00	367.60	57.00	18.00	25.00	415.80	0.74
16	62.00	12.00	26.00	412.80	66.00	8.00	26.00	422.00	0.25
18	51.00	19.00	30.00	386.20	56.00	17.00	27.00	406.00	0.35
20	64.00	13.00	23.00	440.20	67.00	10.00	23.00	434.80	-0.06
23	65.00	15.00	20.00	455.00	62.00	15.00	23.00	438.60	-0.22
26	64.00	10.00	26.00	412.20	64.00	11.00	25.00	422.00	0.13
30	64.00	11.00	25.00	425.80	70.00	11.00	19.00	465.40	0.65
35	71.00	9.00	20.00	462.00	67.00	13.00	20.00	459.40	-0.12
40	64.00	8.00	28.00	404.80	69.00	13.00	18.00	476.00	1.20
50	65.00	14.00	21.00	454.40	74.00	13.00	13.00	511.40	0.96
60	64.00	13.00	23.00	437.20	68.00	13.00	19.00	468.40	0.57
80	70.00	17.00	13.00	510.40	75.00	10.00	15.00	493.40	-0.28
100	72.00	14.00	14.00	505.60	76.00	10.00	14.00	504.40	-0.02

Table A.6: 6♡ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	31.00	8.00	61.00	450.80	28.00	7.00	65.00	383.60	-1.01
2	34.00	4.00	62.00	440.10	35.00	8.00	57.00	513.00	0.57
3	44.00	4.00	52.00	590.70	45.00	4.00	51.00	609.70	0.08
4	38.00	7.00	55.00	554.20	44.00	6.00	50.00	633.60	0.88
5	47.00	4.00	49.00	636.60	51.00	3.00	46.00	706.10	0.91
6	51.00	6.00	43.00	746.70	53.00	6.00	41.00	775.60	0.24
7	48.00	8.00	44.00	724.20	56.00	2.00	42.00	760.60	0.56
8	55.00	5.00	40.00	794.60	57.00	1.00	42.00	768.90	-0.25
9	56.00	4.00	40.00	798.90	57.00	5.00	38.00	820.20	0.12
10	50.00	4.00	46.00	700.10	63.00	4.00	33.00	906.30	2.33
12	60.00	3.00	37.00	848.50	61.00	4.00	35.00	875.70	0.20
14	60.00	3.00	37.00	848.10	66.00	0.00	34.00	894.90	0.53
16	49.00	6.00	45.00	720.10	63.00	1.00	36.00	868.70	1.76
18	65.00	3.00	32.00	928.60	67.00	6.00	27.00	1007.80	0.79
20	64.00	3.00	33.00	913.30	66.00	2.00	32.00	927.20	0.15
23	64.00	4.00	32.00	933.30	72.00	2.00	26.00	1028.40	1.01
26	68.00	2.00	30.00	960.50	66.00	4.00	30.00	957.20	-0.08
30	59.00	6.00	35.00	878.50	75.00	1.00	24.00	1056.20	2.11
35	75.00	4.00	21.00	1101.20	73.00	1.00	26.00	1024.30	-0.66
40	72.00	3.00	25.00	1037.00	73.00	2.00	25.00	1037.60	0.05
50	77.00	2.00	21.00	1097.80	74.00	2.00	24.00	1057.90	-0.30
60	70.00	1.00	29.00	982.40	73.00	2.00	25.00	1038.60	0.57
80	77.00	1.00	22.00	1094.20	74.00	3.00	23.00	1072.90	-0.28
100	73.00	3.00	24.00	1057.00	77.00	0.00	23.00	1072.50	0.19

Table A.7: 6NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	34.00	7.00	59.00	461.40	37.00	5.00	58.00	472.40	0.06
2	38.00	4.00	58.00	493.70	45.00	3.00	52.00	587.40	0.96
3	48.00	3.00	49.00	648.20	45.00	4.00	51.00	607.50	-0.58
4	51.00	3.00	46.00	692.80	51.00	6.00	43.00	738.30	0.58
5	54.00	5.00	41.00	770.70	56.00	3.00	41.00	773.00	0.06
6	56.00	5.00	39.00	812.90	59.00	5.00	36.00	858.80	0.38
7	58.00	5.00	37.00	847.30	64.00	5.00	31.00	942.00	1.18
8	62.00	3.00	35.00	875.80	62.00	5.00	33.00	909.90	0.39
9	63.00	4.00	33.00	916.20	62.00	4.00	34.00	888.20	-0.48
10	70.00	3.00	27.00	1020.60	62.00	2.00	36.00	865.70	-1.80
12	71.00	2.00	27.00	1010.90	66.00	4.00	30.00	951.10	-0.71
14	64.00	2.00	34.00	897.80	67.00	3.00	30.00	953.40	0.52
16	68.00	6.00	26.00	1023.40	75.00	3.00	22.00	1091.90	0.79
18	70.00	3.00	27.00	1001.60	68.00	4.00	28.00	995.90	-0.06
20	69.00	3.00	28.00	994.20	74.00	3.00	23.00	1073.50	0.75
23	74.00	3.00	23.00	1078.50	72.00	4.00	24.00	1053.20	-0.28
26	71.00	3.00	26.00	1023.00	76.00	3.00	21.00	1100.00	0.84
30	71.00	6.00	23.00	1069.90	76.00	3.00	21.00	1107.60	0.48
35	74.00	3.00	23.00	1075.50	69.00	6.00	25.00	1048.10	-0.22
40	73.00	6.00	21.00	1100.70	79.00	3.00	18.00	1155.20	0.66
50	72.00	5.00	23.00	1075.90	80.00	4.00	16.00	1186.00	1.09
60	71.00	4.00	25.00	1040.40	79.00	2.00	19.00	1133.10	1.03
80	76.00	4.00	20.00	1123.10	79.00	3.00	18.00	1152.50	0.33
100	71.00	5.00	24.00	1064.50	78.00	4.00	18.00	1156.20	1.12

Table A.8: 7♡ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	18.00	0.00	82.00	241.80	19.00	0.00	81.00	270.90	0.31
2	21.00	0.00	79.00	304.10	19.00	0.00	81.00	249.90	-0.59
3	30.00	0.00	70.00	535.00	28.00	0.00	72.00	494.80	-0.24
4	33.00	0.00	67.00	603.30	34.00	0.00	66.00	637.40	0.25
5	40.00	0.00	60.00	774.00	30.00	0.00	70.00	536.00	-2.01
6	37.00	0.00	63.00	702.70	45.00	0.00	55.00	902.50	1.87
7	41.00	0.00	59.00	801.10	48.00	0.00	52.00	971.80	1.49
8	47.00	0.00	53.00	948.70	46.00	0.00	54.00	927.60	-0.14
9	43.00	0.00	57.00	857.30	49.00	0.00	51.00	998.90	1.21
10	44.00	0.00	56.00	885.40	50.00	0.00	50.00	1027.00	1.15
12	51.00	0.00	49.00	1038.10	63.00	0.00	37.00	1331.30	2.57
14	50.00	0.00	50.00	1022.00	55.00	0.00	45.00	1145.50	0.97
16	51.00	0.00	49.00	1049.10	57.00	0.00	43.00	1193.70	1.28
18	56.00	0.00	44.00	1169.60	57.00	0.00	43.00	1194.70	0.30
20	56.00	0.00	44.00	1170.60	49.00	0.00	51.00	1005.90	-1.35
23	54.00	0.00	46.00	1121.40	59.00	0.00	41.00	1241.90	1.12
26	63.00	0.00	37.00	1333.30	66.00	0.00	34.00	1405.60	0.66
30	65.00	0.00	35.00	1381.50	61.00	0.00	39.00	1288.10	-0.76
35	64.00	0.00	36.00	1363.40	61.00	0.00	39.00	1292.10	-0.54
40	64.00	0.00	36.00	1364.40	68.00	0.00	32.00	1450.80	0.65
50	62.00	0.00	38.00	1317.20	70.00	0.00	30.00	1504.00	1.61
60	64.00	0.00	36.00	1362.40	66.00	0.00	34.00	1408.60	0.37
80	69.00	0.00	31.00	1477.90	68.00	0.00	32.00	1456.80	-0.10
100	68.00	0.00	32.00	1453.80	69.00	0.00	31.00	1476.90	0.27

Table A.9: 7NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	27.00	0.00	73.00	459.40	31.00	0.00	69.00	541.20	0.85
2	40.00	0.00	60.00	770.00	39.00	0.00	61.00	755.80	-0.04
3	47.00	0.00	53.00	942.40	54.00	0.00	46.00	1124.80	1.49
4	54.00	0.00	46.00	1114.80	62.00	0.00	38.00	1311.40	1.70
5	57.00	0.00	43.00	1175.40	65.00	0.00	35.00	1387.00	1.86
6	55.00	0.00	45.00	1129.00	66.00	0.00	34.00	1390.20	2.26
7	64.00	0.00	36.00	1350.80	65.00	0.00	35.00	1381.00	0.17
8	60.00	0.00	40.00	1266.00	64.00	0.00	36.00	1361.80	0.95
9	62.00	0.00	38.00	1307.40	64.00	0.00	36.00	1351.80	0.38
10	69.00	0.00	31.00	1478.80	70.00	0.00	30.00	1502.00	0.14
12	67.00	0.00	33.00	1427.40	69.00	0.00	31.00	1470.80	0.46
14	72.00	0.00	28.00	1552.40	78.00	0.00	22.00	1691.60	1.16
16	68.00	0.00	32.00	1457.60	74.00	0.00	26.00	1584.80	0.82
18	70.00	0.00	30.00	1505.00	73.00	0.00	27.00	1575.60	0.70
20	74.00	0.00	26.00	1594.80	74.00	0.00	26.00	1591.80	0.04
23	77.00	0.00	23.00	1660.40	78.00	0.00	22.00	1693.60	0.37
26	72.00	0.00	28.00	1556.40	74.00	0.00	26.00	1606.80	0.50
30	73.00	0.00	27.00	1571.60	79.00	0.00	21.00	1709.80	1.17
35	73.00	0.00	27.00	1566.60	74.00	0.00	26.00	1599.80	0.35
40	70.00	0.00	30.00	1499.00	75.00	0.00	25.00	1610.00	0.79
50	75.00	0.00	25.00	1620.00	81.00	0.00	19.00	1767.20	1.31
60	76.00	0.00	24.00	1642.20	81.00	0.00	19.00	1759.20	0.87
80	78.00	0.00	22.00	1694.60	81.00	0.00	19.00	1761.20	0.49
100	76.00	0.00	24.00	1645.20	81.00	0.00	19.00	1761.20	0.88

Table A.10: Aggregated Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	31.01	26.23	42.76	176.63	27.49	28.65	43.86	165.62	-0.16
2	32.86	25.54	41.61	190.78	31.80	25.40	42.80	187.13	-0.12
3	38.18	26.82	34.99	240.54	37.28	25.72	37.00	229.32	-0.19
4	40.00	26.05	33.95	248.60	40.20	24.36	35.44	250.44	0.03
5	43.07	23.57	33.36	258.65	39.86	25.98	34.16	259.37	-0.04
6	46.10	24.15	29.74	287.64	46.20	24.60	29.20	301.26	0.19
7	47.30	25.59	27.11	301.42	50.68	21.75	27.58	300.92	-0.01
8	48.70	24.56	26.75	312.85	49.02	22.46	28.52	297.95	-0.24
9	50.05	22.71	27.24	307.39	49.47	23.66	26.87	310.13	0.06
10	47.14	23.31	29.55	285.56	54.89	22.22	22.89	331.37	0.74
12	52.89	20.09	27.01	309.45	56.50	19.89	23.61	330.41	0.33
14	53.41	22.34	24.25	327.79	56.80	22.27	20.93	348.28	0.34
16	55.54	21.86	22.60	331.52	59.91	19.34	20.76	357.02	0.37
18	56.22	20.85	22.93	338.50	59.39	19.11	21.50	355.85	0.27
20	58.69	20.60	20.71	356.69	57.94	19.35	22.71	344.94	-0.19
23	60.13	20.92	18.95	370.49	63.15	18.51	18.34	374.53	0.05
26	63.37	16.77	19.86	366.93	63.59	18.64	17.77	382.60	0.25
30	57.79	20.31	21.90	351.57	64.76	18.38	16.86	382.65	0.48
35	66.30	18.43	15.27	400.55	69.02	16.31	14.67	402.17	0.07
40	63.74	17.51	18.75	372.03	66.47	17.18	16.35	398.04	0.38
50	66.86	17.93	15.21	402.96	65.41	18.13	16.45	388.18	-0.22
60	66.37	17.30	16.33	390.16	69.67	16.09	14.24	401.76	0.19
80	68.08	16.75	15.17	402.36	70.93	16.64	12.43	416.05	0.25
100	68.43	17.71	13.86	414.13	71.76	16.15	12.09	418.82	0.11

Comparison of our method with and without spot card rellocation

Table A.11: 1♡ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	31.00	45.00	24.00	54.00	35.00	43.00	22.00	62.30	0.21
2	29.00	42.00	29.00	49.60	31.00	43.00	26.00	56.00	0.17
3	40.00	38.00	22.00	68.60	34.00	39.00	27.00	53.50	-0.40
4	40.00	35.00	25.00	63.40	36.00	39.00	25.00	54.60	-0.20
5	35.00	37.00	28.00	47.30	36.00	43.00	21.00	68.80	0.59
6	37.00	38.00	25.00	62.00	44.00	35.00	21.00	69.90	0.22
7	49.00	33.00	18.00	76.10	45.00	34.00	21.00	70.70	-0.16
8	44.00	34.00	22.00	68.40	48.00	32.00	20.00	70.90	0.04
9	43.00	38.00	19.00	75.10	49.00	31.00	20.00	75.70	0.03
10	53.00	34.00	13.00	91.90	51.00	31.00	18.00	76.80	-0.41
12	53.00	32.00	15.00	82.80	53.00	27.00	20.00	76.30	-0.17
14	52.00	32.00	16.00	79.00	51.00	32.00	17.00	83.80	0.13
16	54.00	29.00	17.00	82.30	57.00	31.00	12.00	93.00	0.29
18	55.00	28.00	17.00	85.60	60.00	27.00	13.00	93.40	0.21
20	57.00	27.00	16.00	85.10	67.00	24.00	9.00	99.50	0.41
23	59.00	27.00	14.00	88.70	57.00	28.00	15.00	87.90	-0.02
26	57.00	27.00	16.00	85.90	65.00	25.00	10.00	100.00	0.39
30	58.00	30.00	12.00	92.60	62.00	23.00	15.00	90.40	-0.07
35	66.00	21.00	13.00	93.50	62.00	25.00	13.00	95.50	0.04
40	61.00	23.00	16.00	89.90	64.00	20.00	16.00	91.40	0.05
50	58.00	31.00	11.00	97.10	72.00	16.00	12.00	100.70	0.14
60	71.00	20.00	9.00	103.30	69.00	20.00	11.00	101.60	-0.04
80	69.00	22.00	9.00	103.90	78.00	12.00	10.00	105.50	0.06
100	68.00	22.00	10.00	99.90	74.00	17.00	9.00	104.50	0.13

Table A.12: 1NT results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	33.00	29.00	38.00	14.00	33.00	25.00	42.00	2.60	-0.21
2	45.00	24.00	31.00	44.80	33.00	35.00	32.00	28.80	-0.37
3	43.00	21.00	36.00	18.40	41.00	29.00	30.00	46.40	0.62
4	49.00	22.00	29.00	45.60	44.00	34.00	22.00	59.20	0.30
5	53.00	25.00	22.00	63.50	51.00	32.00	17.00	73.80	0.23
6	54.00	24.00	22.00	64.90	58.00	24.00	18.00	74.80	0.28
7	55.00	23.00	22.00	62.50	53.00	32.00	15.00	79.90	0.43
8	55.00	29.00	16.00	77.60	57.00	19.00	24.00	63.80	-0.37
9	59.00	24.00	17.00	82.40	64.00	20.00	16.00	80.20	-0.04
10	61.00	24.00	15.00	86.10	60.00	22.00	18.00	76.60	-0.22
12	67.00	23.00	10.00	92.40	60.00	24.00	16.00	84.30	-0.21
14	69.00	18.00	13.00	92.40	65.00	23.00	12.00	88.70	-0.08
16	65.00	24.00	11.00	96.60	67.00	19.00	14.00	88.50	-0.21
18	68.00	18.00	14.00	86.30	70.00	22.00	8.00	101.00	0.35
20	58.00	22.00	20.00	80.70	69.00	15.00	16.00	87.40	0.21
23	69.00	20.00	11.00	93.50	63.00	23.00	14.00	87.80	-0.16
26	72.00	15.00	13.00	91.30	68.00	19.00	13.00	91.50	0.02
30	70.00	19.00	11.00	94.90	74.00	16.00	10.00	99.40	0.14
35	71.00	20.00	9.00	100.40	72.00	20.00	8.00	100.70	-0.01
40	70.00	18.00	12.00	93.30	79.00	14.00	7.00	108.70	0.39
50	68.00	23.00	9.00	100.20	73.00	18.00	9.00	103.30	0.09
60	75.00	16.00	9.00	101.80	69.00	20.00	11.00	95.30	-0.20
80	72.00	18.00	10.00	100.10	77.00	15.00	8.00	106.40	0.18
100	78.00	15.00	7.00	109.20	79.00	17.00	4.00	113.70	0.11

Table A.13: 3NT results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	23.00	26.00	51.00	216.70	33.00	25.00	42.00	279.50	0.87
2	34.00	24.00	42.00	295.50	41.00	24.00	35.00	346.50	0.76
3	41.00	24.00	35.00	346.10	47.00	22.00	31.00	369.70	0.29
4	47.00	19.00	34.00	364.70	41.00	25.00	34.00	364.40	0.01
5	50.00	24.00	26.00	429.60	51.00	20.00	29.00	404.40	-0.39
6	58.00	20.00	22.00	454.60	55.00	15.00	30.00	394.60	-1.03
7	55.00	16.00	29.00	399.60	57.00	12.00	31.00	387.30	-0.18
8	57.00	16.00	27.00	419.40	57.00	18.00	25.00	433.80	0.27
9	58.00	18.00	24.00	447.20	58.00	17.00	25.00	434.10	-0.26
10	56.00	22.00	22.00	449.40	60.00	23.00	17.00	497.40	0.80
12	66.00	16.00	18.00	489.70	62.00	16.00	22.00	450.60	-0.61
14	64.00	17.00	19.00	473.00	69.00	16.00	15.00	514.20	0.74
16	67.00	12.00	21.00	470.20	65.00	19.00	16.00	499.20	0.53
18	66.00	14.00	20.00	470.60	66.00	13.00	21.00	460.90	-0.21
20	63.00	19.00	18.00	488.30	66.00	16.00	18.00	489.50	0.01
23	69.00	15.00	16.00	505.70	75.00	15.00	10.00	547.40	0.65
26	75.00	13.00	12.00	534.30	72.00	12.00	16.00	506.70	-0.45
30	67.00	15.00	18.00	486.10	66.00	13.00	21.00	464.80	-0.23
35	74.00	11.00	15.00	517.50	74.00	12.00	14.00	519.10	0.02
40	69.00	13.00	18.00	494.40	76.00	13.00	11.00	545.60	0.86
50	75.00	11.00	14.00	522.00	79.00	11.00	10.00	552.90	0.51
60	73.00	17.00	10.00	551.10	81.00	8.00	11.00	547.50	-0.02
80	72.00	14.00	14.00	522.40	81.00	8.00	11.00	544.80	0.40
100	74.00	15.00	11.00	542.10	77.00	11.00	12.00	537.60	-0.05

Table A.14: 4♡ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	26.00	23.00	51.00	236.40	25.00	18.00	57.00	187.50	-0.83
2	29.00	18.00	53.00	215.60	29.00	29.00	42.00	309.50	1.60
3	30.00	25.00	45.00	285.60	31.00	18.00	51.00	242.80	-0.62
4	33.00	25.00	42.00	313.70	31.00	22.00	47.00	274.40	-0.66
5	32.00	25.00	43.00	298.30	40.00	22.00	38.00	339.70	0.67
6	43.00	22.00	35.00	369.80	40.00	23.00	37.00	349.50	-0.41
7	48.00	21.00	31.00	401.00	47.00	20.00	33.00	381.90	-0.38
8	45.00	21.00	34.00	370.70	51.00	24.00	25.00	446.10	1.25
9	46.00	20.00	34.00	375.70	52.00	23.00	25.00	446.20	1.16
10	52.00	17.00	31.00	396.50	54.00	17.00	29.00	417.50	0.37
12	50.00	15.00	35.00	374.60	54.00	23.00	23.00	457.90	1.25
14	51.00	24.00	25.00	439.50	57.00	14.00	29.00	422.50	-0.22
16	57.00	21.00	22.00	468.00	59.00	15.00	26.00	441.00	-0.39
18	56.00	18.00	26.00	440.40	60.00	17.00	23.00	463.70	0.38
20	51.00	18.00	31.00	403.50	55.00	22.00	23.00	465.10	1.01
23	60.00	17.00	23.00	460.70	63.00	18.00	19.00	491.10	0.47
26	59.00	21.00	20.00	487.00	64.00	18.00	18.00	500.70	0.22
30	65.00	15.00	20.00	483.50	64.00	19.00	17.00	511.50	0.49
35	68.00	19.00	13.00	538.60	64.00	18.00	18.00	498.30	-0.70
40	67.00	19.00	14.00	531.80	71.00	14.00	15.00	525.40	-0.05
50	61.00	14.00	25.00	450.60	70.00	17.00	13.00	540.60	1.47
60	64.00	16.00	20.00	491.20	74.00	13.00	13.00	543.10	0.87
80	70.00	18.00	12.00	546.80	76.00	14.00	10.00	563.70	0.30
100	70.00	17.00	13.00	540.80	75.00	15.00	10.00	564.70	0.39

Table A.15: 5♢ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	25.00	15.00	60.00	150.20	25.00	19.00	56.00	179.60	0.43
2	31.00	12.00	57.00	178.40	28.00	14.00	58.00	162.60	-0.29
3	32.00	12.00	56.00	190.20	36.00	18.00	46.00	255.60	1.13
4	37.00	16.00	47.00	260.60	40.00	12.00	48.00	236.00	-0.45
5	38.00	16.00	46.00	259.40	46.00	12.00	42.00	292.80	0.73
6	48.00	14.00	38.00	325.80	44.00	12.00	44.00	275.60	-0.89
7	46.00	14.00	40.00	295.80	48.00	15.00	37.00	330.40	0.54
8	49.00	14.00	37.00	331.80	47.00	12.00	41.00	304.80	-0.44
9	48.00	15.00	37.00	325.40	54.00	17.00	29.00	392.20	1.14
10	57.00	13.00	30.00	384.00	57.00	15.00	28.00	399.80	0.22
12	53.00	14.00	33.00	360.80	60.00	12.00	28.00	400.20	0.68
14	57.00	18.00	25.00	415.80	69.00	12.00	19.00	466.00	0.82
16	66.00	8.00	26.00	422.00	60.00	13.00	27.00	399.20	-0.46
18	56.00	17.00	27.00	406.00	62.00	14.00	24.00	427.20	0.35
20	67.00	10.00	23.00	434.80	69.00	11.00	20.00	457.00	0.37
23	62.00	15.00	23.00	438.60	67.00	10.00	23.00	437.80	0.01
26	64.00	11.00	25.00	422.00	65.00	13.00	22.00	441.20	0.27
30	70.00	11.00	19.00	465.40	70.00	14.00	16.00	486.60	0.39
35	67.00	13.00	20.00	459.40	71.00	13.00	16.00	489.80	0.54
40	69.00	13.00	18.00	476.00	71.00	11.00	18.00	473.80	-0.02
50	74.00	13.00	13.00	511.40	72.00	12.00	16.00	486.20	-0.44
60	68.00	13.00	19.00	468.40	72.00	12.00	16.00	492.60	0.41
80	75.00	10.00	15.00	493.40	70.00	14.00	16.00	488.20	-0.09
100	76.00	10.00	14.00	504.40	76.00	14.00	10.00	534.00	0.50

Table A.16: 6♡ results

No of	No Relocation				Relocation				IMP
Samples	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	Difference
1	28.00	7.00	65.00	383.60	30.00	6.00	64.00	417.20	0.69
2	35.00	8.00	57.00	513.00	32.00	2.00	66.00	361.50	-1.73
3	45.00	4.00	51.00	609.70	42.00	5.00	53.00	585.10	-0.10
4	44.00	6.00	50.00	633.60	44.00	7.00	49.00	642.70	-0.05
5	51.00	3.00	46.00	706.10	48.00	4.00	48.00	667.20	-0.45
6	53.00	6.00	41.00	775.60	53.00	3.00	44.00	726.70	-0.46
7	56.00	2.00	42.00	760.60	53.00	4.00	43.00	746.30	-0.19
8	57.00	1.00	42.00	768.90	58.00	5.00	37.00	842.50	0.74
9	57.00	5.00	38.00	820.20	56.00	2.00	42.00	768.20	-0.44
10	63.00	4.00	33.00	906.30	61.00	2.00	37.00	845.10	-0.72
12	61.00	4.00	35.00	875.70	61.00	4.00	35.00	879.10	0.07
14	66.00	0.00	34.00	894.90	66.00	3.00	31.00	944.50	0.51
16	63.00	1.00	36.00	868.70	68.00	5.00	27.00	1002.80	1.38
18	67.00	6.00	27.00	1007.80	63.00	3.00	34.00	894.60	-1.12
20	66.00	2.00	32.00	927.20	69.00	1.00	30.00	964.10	0.50
23	72.00	2.00	26.00	1028.40	66.00	6.00	28.00	985.50	-0.54
26	66.00	4.00	30.00	957.20	74.00	1.00	25.00	1042.90	1.04
30	75.00	1.00	24.00	1056.20	74.00	5.00	21.00	1103.20	0.50
35	73.00	1.00	26.00	1024.30	76.00	3.00	21.00	1106.80	0.84
40	73.00	2.00	25.00	1037.60	76.00	2.00	22.00	1092.80	0.65
50	74.00	2.00	24.00	1057.90	73.00	1.00	26.00	1029.60	-0.24
60	73.00	2.00	25.00	1038.60	80.00	1.00	19.00	1135.00	1.01
80	74.00	3.00	23.00	1072.90	78.00	2.00	20.00	1122.40	0.57
100	77.00	0.00	23.00	1072.50	77.00	2.00	21.00	1108.10	0.39

Table A.17: 6NT results

No of	No Relocation				Relocation				IMP
Samples	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	Difference
1	37.00	5.00	58.00	472.40	34.00	5.00	61.00	437.90	-0.27
2	45.00	3.00	52.00	587.40	44.00	2.00	54.00	562.50	-0.33
3	45.00	4.00	51.00	607.50	47.00	5.00	48.00	661.00	0.49
4	51.00	6.00	43.00	738.30	57.00	4.00	39.00	799.20	0.49
5	56.00	3.00	41.00	773.00	52.00	3.00	45.00	713.10	-0.51
6	59.00	5.00	36.00	858.80	56.00	5.00	39.00	811.20	-0.48
7	64.00	5.00	31.00	942.00	65.00	4.00	31.00	938.70	-0.13
8	62.00	5.00	33.00	909.90	63.00	5.00	32.00	923.00	0.16
9	62.00	4.00	34.00	888.20	70.00	3.00	27.00	1015.90	1.33
10	62.00	2.00	36.00	865.70	69.00	5.00	26.00	1024.00	1.62
12	66.00	4.00	30.00	951.10	67.00	4.00	29.00	973.50	0.37
14	67.00	3.00	30.00	953.40	67.00	4.00	29.00	975.50	0.33
16	75.00	3.00	22.00	1091.90	76.00	4.00	20.00	1113.40	0.09
18	68.00	4.00	28.00	995.90	76.00	3.00	21.00	1110.70	1.24
20	74.00	3.00	23.00	1073.50	76.00	2.00	22.00	1093.20	0.25
23	72.00	4.00	24.00	1053.20	73.00	3.00	24.00	1057.10	0.14
26	76.00	3.00	21.00	1100.00	79.00	1.00	20.00	1121.00	0.33
30	76.00	3.00	21.00	1107.60	77.00	3.00	20.00	1125.70	0.16
35	69.00	6.00	25.00	1048.10	75.00	3.00	22.00	1093.90	0.46
40	79.00	3.00	18.00	1155.20	79.00	4.00	17.00	1168.60	0.11
50	80.00	4.00	16.00	1186.00	73.00	4.00	23.00	1076.50	-1.07
60	79.00	2.00	19.00	1133.10	79.00	3.00	18.00	1151.80	0.22
80	79.00	3.00	18.00	1152.50	81.00	3.00	16.00	1185.30	0.30
100	78.00	4.00	18.00	1156.20	75.00	4.00	21.00	1099.70	-0.65

Table A.18: 7♡ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	19.00	0.00	81.00	270.90	19.00	0.00	81.00	284.90	0.29
2	19.00	0.00	81.00	249.90	26.00	0.00	74.00	425.60	1.62
3	28.00	0.00	72.00	494.80	33.00	0.00	67.00	609.30	0.82
4	34.00	0.00	66.00	637.40	35.00	0.00	65.00	658.50	0.33
5	30.00	0.00	70.00	536.00	35.00	0.00	65.00	657.50	1.06
6	45.00	0.00	55.00	902.50	40.00	0.00	60.00	785.00	-0.98
7	48.00	0.00	52.00	971.80	44.00	0.00	56.00	877.40	-0.85
8	46.00	0.00	54.00	927.60	42.00	0.00	58.00	840.20	-0.70
9	49.00	0.00	51.00	998.90	50.00	0.00	50.00	1030.00	0.42
10	50.00	0.00	50.00	1027.00	51.00	0.00	49.00	1045.10	0.02
12	63.00	0.00	37.00	1331.30	55.00	0.00	45.00	1139.50	-1.62
14	55.00	0.00	45.00	1145.50	61.00	0.00	39.00	1284.10	1.24
16	57.00	0.00	43.00	1193.70	54.00	0.00	46.00	1126.40	-0.56
18	57.00	0.00	43.00	1194.70	57.00	0.00	43.00	1193.70	-0.03
20	49.00	0.00	51.00	1005.90	55.00	0.00	45.00	1146.50	1.21
23	59.00	0.00	41.00	1241.90	62.00	0.00	38.00	1309.20	0.54
26	66.00	0.00	34.00	1405.60	66.00	0.00	34.00	1406.60	0.06
30	61.00	0.00	39.00	1288.10	65.00	0.00	35.00	1384.50	0.85
35	61.00	0.00	39.00	1292.10	65.00	0.00	35.00	1385.50	0.82
40	68.00	0.00	32.00	1450.80	61.00	0.00	39.00	1294.10	-1.28
50	70.00	0.00	30.00	1504.00	66.00	0.00	34.00	1407.60	-0.89
60	66.00	0.00	34.00	1408.60	71.00	0.00	29.00	1523.10	0.97
80	68.00	0.00	32.00	1456.80	71.00	0.00	29.00	1526.10	0.56
100	69.00	0.00	31.00	1476.90	72.00	0.00	28.00	1550.20	0.68

Table A.19: 7NT results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	31.00	0.00	69.00	541.20	39.00	0.00	61.00	738.80	1.64
2	39.00	0.00	61.00	755.80	40.00	0.00	60.00	776.00	0.02
3	54.00	0.00	46.00	1124.80	47.00	0.00	53.00	948.40	-1.60
4	62.00	0.00	38.00	1311.40	55.00	0.00	45.00	1142.00	-1.47
5	65.00	0.00	35.00	1387.00	57.00	0.00	43.00	1185.40	-1.80
6	66.00	0.00	34.00	1390.20	66.00	0.00	34.00	1406.20	0.36
7	65.00	0.00	35.00	1381.00	62.00	0.00	38.00	1310.40	-0.61
8	64.00	0.00	36.00	1361.80	67.00	0.00	33.00	1429.40	0.53
9	64.00	0.00	36.00	1351.80	73.00	0.00	27.00	1569.60	1.61
10	70.00	0.00	30.00	1502.00	67.00	0.00	33.00	1426.40	-0.60
12	69.00	0.00	31.00	1470.80	67.00	0.00	33.00	1420.40	-0.59
14	78.00	0.00	22.00	1691.60	77.00	0.00	23.00	1664.40	-0.39
16	74.00	0.00	26.00	1584.80	75.00	0.00	25.00	1611.00	0.35
18	73.00	0.00	27.00	1575.60	76.00	0.00	24.00	1648.20	0.60
20	74.00	0.00	26.00	1591.80	75.00	0.00	25.00	1619.00	0.22
23	78.00	0.00	22.00	1693.60	76.00	0.00	24.00	1629.20	-0.66
26	74.00	0.00	26.00	1606.80	80.00	0.00	20.00	1745.00	1.14
30	79.00	0.00	21.00	1709.80	75.00	0.00	25.00	1623.00	-0.68
35	74.00	0.00	26.00	1599.80	76.00	0.00	24.00	1637.20	0.21
40	75.00	0.00	25.00	1610.00	76.00	0.00	24.00	1636.20	0.39
50	81.00	0.00	19.00	1767.20	81.00	0.00	19.00	1767.20	-0.10
60	81.00	0.00	19.00	1759.20	80.00	0.00	20.00	1740.00	-0.15
80	81.00	0.00	19.00	1761.20	84.00	0.00	16.00	1835.80	0.66
100	81.00	0.00	19.00	1761.20	83.00	0.00	17.00	1814.60	0.54

Table A.20: Aggregated Results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	27.49	28.65	43.86	165.62	30.47	26.40	43.13	170.29	0.06
2	31.80	25.40	42.80	187.13	32.47	29.44	38.10	215.30	0.51
3	37.28	25.72	37.00	229.32	37.00	24.81	38.18	224.35	-0.10
4	40.20	24.36	35.44	250.44	37.15	26.58	36.27	236.87	-0.25
5	39.86	25.98	34.16	259.37	42.86	26.42	30.72	273.62	0.33
6	46.20	24.60	29.20	301.26	46.77	22.68	30.54	278.92	-0.34
7	50.68	21.75	27.58	300.92	49.41	21.87	28.72	293.86	-0.13
8	49.02	22.46	28.52	297.95	51.92	22.40	25.69	323.26	0.39
9	49.47	23.66	26.87	310.13	53.77	21.90	24.33	330.61	0.35
10	54.89	22.22	22.89	331.37	55.51	21.43	23.06	340.22	0.11
12	56.50	19.89	23.61	330.41	56.74	20.59	22.67	345.13	0.21
14	56.80	22.27	20.93	348.28	59.81	19.40	20.79	360.71	0.22
16	59.91	19.34	20.76	357.02	60.93	19.86	19.21	363.46	0.11
18	59.39	19.11	21.50	355.85	62.41	18.24	19.35	359.69	0.12
20	57.94	19.35	22.71	344.94	63.85	18.45	17.70	372.26	0.49
23	63.15	18.51	18.34	374.53	64.17	19.16	16.66	388.50	0.22
26	63.59	18.64	17.77	382.60	66.99	17.30	15.72	391.90	0.17
30	64.76	18.38	16.86	382.65	65.75	17.27	16.98	391.08	0.15
35	69.02	16.31	14.67	402.17	67.36	17.50	15.14	400.03	-0.08
40	66.47	17.18	16.35	398.04	70.84	14.44	14.72	410.00	0.23
50	65.41	18.13	16.45	388.18	72.94	14.04	13.02	415.75	0.50
60	69.67	16.09	14.24	401.76	73.74	13.66	12.60	423.00	0.31
80	70.93	16.64	12.43	416.05	77.32	11.40	11.27	429.54	0.23
100	71.76	16.15	12.09	418.82	75.60	13.91	10.49	430.94	0.21

Appendix B

RESULT TABLES AGAINST OPTIMIZED PERFECT DEFENDER

Comparison of the Monte-Carlo method and our method

Table B.1: 1♡ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	21.00	47.00	32.00	29.80	37.00	38.00	25.00	58.20	0.79
5	38.00	38.00	24.00	62.40	39.00	35.00	26.00	61.60	-0.02
10	48.00	29.00	23.00	68.80	55.00	27.00	18.00	78.40	0.26
20	44.00	33.00	23.00	74.00	55.00	29.00	16.00	87.70	0.38
35	58.00	29.00	13.00	93.70	66.00	21.00	13.00	95.90	0.07
50	62.00	23.00	15.00	90.00	67.00	20.00	13.00	93.80	0.12
80	63.00	26.00	11.00	96.60	68.00	20.00	12.00	97.90	0.04
100	58.00	24.00	18.00	85.60	73.00	18.00	9.00	105.20	0.54

Table B.2: 1NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	25.00	35.00	40.00	2.20	36.00	33.00	31.00	27.60	0.66
5	53.00	28.00	19.00	70.90	52.00	26.00	22.00	60.30	-0.27
10	57.00	29.00	14.00	82.40	65.00	20.00	15.00	76.30	-0.11
20	60.00	23.00	17.00	78.50	63.00	24.00	13.00	93.60	0.39
35	67.00	25.00	8.00	103.20	72.00	19.00	9.00	100.10	-0.07
50	67.00	24.00	9.00	101.20	62.00	27.00	11.00	94.50	-0.14
80	71.00	15.00	14.00	92.60	68.00	25.00	7.00	103.10	0.25
100	69.00	21.00	10.00	98.20	68.00	24.00	8.00	100.70	0.06

Table B.3: 3NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	24.00	24.00	52.00	200.90	35.00	19.00	46.00	248.50	0.65
5	52.00	16.00	32.00	375.70	58.00	16.00	26.00	425.00	0.82
10	54.00	18.00	28.00	409.90	57.00	18.00	25.00	433.50	0.44
20	55.00	23.00	22.00	455.20	63.00	17.00	20.00	462.90	0.21
35	65.00	18.00	17.00	498.30	67.00	21.00	12.00	534.00	0.58
50	71.00	16.00	13.00	522.60	66.00	22.00	12.00	532.00	0.15
80	67.00	17.00	16.00	504.50	74.00	14.00	12.00	535.80	0.58
100	68.00	19.00	13.00	526.90	70.00	16.00	14.00	519.90	-0.10

Table B.4: 4♡ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	18.00	20.00	62.00	130.40	26.00	23.00	51.00	233.10	1.94
5	39.00	19.00	42.00	301.70	37.00	22.00	41.00	318.10	0.34
10	45.00	18.00	37.00	353.50	45.00	24.00	31.00	397.20	0.67
20	48.00	25.00	27.00	430.20	63.00	19.00	18.00	499.70	1.16
35	63.00	16.00	21.00	479.00	65.00	18.00	17.00	506.30	0.40
50	60.00	17.00	23.00	461.40	67.00	17.00	16.00	517.80	1.01
80	61.00	16.00	23.00	465.00	67.00	19.00	14.00	531.80	1.12
100	58.00	22.00	20.00	486.10	66.00	15.00	19.00	497.40	0.20

Table B.5: 5♢ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	14.00	10.00	76.00	5.80	25.00	13.00	62.00	127.40	2.14
5	38.00	14.00	48.00	251.40	50.00	12.00	38.00	317.00	1.09
10	52.00	17.00	31.00	374.20	52.00	11.00	37.00	336.60	-0.60
20	60.00	9.00	31.00	378.40	68.00	9.00	23.00	436.80	1.03
35	66.00	12.00	22.00	448.00	72.00	9.00	19.00	467.00	0.31
50	76.00	5.00	19.00	468.00	69.00	11.00	20.00	460.80	-0.10
80	68.00	10.00	22.00	440.80	77.00	9.00	14.00	504.00	1.07
100	74.00	6.00	20.00	462.40	70.00	10.00	20.00	459.00	-0.10

Table B.6: 6♡ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	14.00	6.00	80.00	121.30	31.00	6.00	63.00	423.50	3.74
5	48.00	5.00	47.00	679.80	54.00	5.00	41.00	774.00	1.08
10	66.00	5.00	29.00	970.50	60.00	3.00	37.00	845.10	-1.34
20	68.00	3.00	29.00	971.50	69.00	4.00	27.00	1011.40	0.47
35	72.00	3.00	25.00	1036.30	76.00	3.00	21.00	1097.20	0.63
50	76.00	1.00	23.00	1061.50	69.00	4.00	27.00	1004.40	-0.66
80	77.00	3.00	20.00	1111.50	78.00	3.00	19.00	1137.10	0.33
100	75.00	3.00	22.00	1083.90	77.00	4.00	19.00	1134.80	0.61

Table B.7: 6NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	25.00	1.00	74.00	204.50	36.00	8.00	56.00	495.00	2.73
5	53.00	8.00	39.00	805.00	55.00	3.00	42.00	756.40	-0.52
10	62.00	7.00	31.00	944.50	61.00	5.00	34.00	903.50	-0.52
20	64.00	7.00	29.00	981.60	72.00	4.00	24.00	1042.20	0.69
35	67.00	6.00	27.00	1009.70	71.00	5.00	23.00	1071.00	0.53
50	65.00	6.00	29.00	983.90	73.00	6.00	21.00	1101.40	1.25
80	68.00	6.00	26.00	1020.10	75.00	6.00	19.00	1132.20	1.22
100	69.00	5.00	26.00	1031.40	76.00	5.00	19.00	1134.50	1.21

Table B.8: 7♡ Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	9.00	0.00	91.00	-22.10	26.00	0.00	74.00	437.60	4.59
5	35.00	0.00	65.00	661.50	41.00	0.00	59.00	807.10	1.27
10	47.00	0.00	53.00	951.70	55.00	0.00	45.00	1145.50	1.84
20	58.00	0.00	42.00	1213.80	59.00	0.00	41.00	1238.90	0.36
35	63.00	0.00	37.00	1334.30	62.00	0.00	38.00	1309.20	-0.19
50	79.00	0.00	21.00	1709.90	72.00	0.00	28.00	1551.20	-1.32
80	75.00	0.00	25.00	1612.50	76.00	0.00	24.00	1638.60	0.23
100	73.00	0.00	27.00	1570.30	73.00	0.00	27.00	1566.30	-0.05

Table B.9: 7NT Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	17.00	0.00	83.00	167.40	36.00	0.00	64.00	656.20	4.32
5	59.00	0.00	41.00	1218.80	56.00	0.00	44.00	1151.20	-0.77
10	61.00	0.00	39.00	1271.20	66.00	0.00	34.00	1386.20	0.85
20	72.00	0.00	28.00	1538.40	75.00	0.00	25.00	1616.00	0.72
35	64.00	0.00	36.00	1335.80	80.00	0.00	20.00	1738.00	3.42
50	69.00	0.00	31.00	1465.80	79.00	0.00	21.00	1693.80	1.93
80	79.00	0.00	21.00	1708.80	80.00	0.00	20.00	1721.00	-0.02
100	76.00	0.00	24.00	1646.20	79.00	0.00	21.00	1707.80	0.40

Table B.10: Aggregated Results

No of Samples	Monte Carlo				Our Method				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	20.03	28.19	51.79	93.35	32.07	25.45	42.48	173.53	1.39
5	42.87	23.34	33.79	250.00	45.03	22.99	31.98	274.97	0.39
10	50.45	21.11	28.44	303.04	53.47	20.92	25.62	313.84	0.24
20	51.38	24.05	24.57	337.71	61.36	20.13	18.51	371.14	0.62
35	63.00	19.73	17.26	377.60	67.43	17.83	14.74	398.20	0.31
50	65.67	17.11	17.22	382.43	66.73	18.30	14.97	395.90	0.28
80	65.20	17.70	17.11	381.52	70.27	16.89	12.83	414.33	0.57
100	63.24	19.37	17.39	388.73	70.05	15.63	14.32	399.28	0.23

Comparison of our method with and without spot card rellocation

Table B.11: 1♡ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	37.00	38.00	25.00	58.20	29.00	48.00	23.00	57.70	-0.04
5	39.00	35.00	26.00	61.60	46.00	35.00	19.00	77.40	0.45
10	55.00	27.00	18.00	78.40	50.00	31.00	19.00	78.50	0.00
20	55.00	29.00	16.00	87.70	66.00	21.00	13.00	97.00	0.28
35	66.00	21.00	13.00	95.90	62.00	25.00	13.00	94.60	-0.04
50	67.00	20.00	13.00	93.80	66.00	20.00	14.00	94.90	0.02
80	68.00	20.00	12.00	97.90	68.00	19.00	13.00	98.30	0.02
100	73.00	18.00	9.00	105.20	68.00	20.00	12.00	97.90	-0.21

Table B.12: 1NT results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	36.00	33.00	31.00	27.60	32.00	31.00	37.00	14.10	-0.36
5	52.00	26.00	22.00	60.30	57.00	25.00	18.00	71.80	0.30
10	65.00	20.00	15.00	76.30	57.00	28.00	15.00	81.60	0.06
20	63.00	24.00	13.00	93.60	63.00	25.00	12.00	90.20	-0.07
35	72.00	19.00	9.00	100.10	74.00	17.00	9.00	101.00	0.05
50	62.00	27.00	11.00	94.50	67.00	23.00	10.00	98.60	0.09
80	68.00	25.00	7.00	103.10	64.00	27.00	9.00	101.60	-0.04
100	68.00	24.00	8.00	100.70	70.00	23.00	7.00	103.40	0.06

Table B.13: 3NT results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	35.00	19.00	46.00	248.50	34.00	18.00	48.00	240.20	0.08
5	58.00	16.00	26.00	425.00	49.00	18.00	33.00	360.40	-1.08
10	57.00	18.00	25.00	433.50	64.00	16.00	20.00	472.30	0.60
20	63.00	17.00	20.00	462.90	68.00	16.00	16.00	499.30	0.63
35	67.00	21.00	12.00	534.00	67.00	18.00	15.00	513.20	-0.30
50	66.00	22.00	12.00	532.00	71.00	15.00	14.00	518.50	-0.14
80	74.00	14.00	12.00	535.80	74.00	14.00	12.00	537.10	0.02
100	70.00	16.00	14.00	519.90	68.00	16.00	16.00	503.90	-0.28

Table B.14: 4♡ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	26.00	23.00	51.00	233.10	28.00	19.00	53.00	227.30	0.00
5	37.00	22.00	41.00	318.10	46.00	17.00	37.00	352.70	0.63
10	45.00	24.00	31.00	397.20	52.00	24.00	24.00	452.10	0.99
20	63.00	19.00	18.00	499.70	58.00	17.00	25.00	445.40	-0.90
35	65.00	18.00	17.00	506.30	62.00	19.00	19.00	492.90	-0.25
50	67.00	17.00	16.00	517.80	69.00	18.00	13.00	537.70	0.31
80	67.00	19.00	14.00	531.80	67.00	16.00	17.00	507.90	-0.41
100	66.00	15.00	19.00	497.40	68.00	15.00	17.00	506.80	0.12

Table B.15: 5♢ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	25.00	13.00	62.00	127.40	21.00	16.00	63.00	121.40	-0.01
5	50.00	12.00	38.00	317.00	42.00	15.00	43.00	275.20	-0.67
10	52.00	11.00	37.00	336.60	60.00	6.00	34.00	360.60	0.41
20	68.00	9.00	23.00	436.80	67.00	14.00	19.00	465.60	0.40
35	72.00	9.00	19.00	467.00	72.00	8.00	20.00	463.00	-0.06
50	69.00	11.00	20.00	460.80	72.00	14.00	14.00	502.60	0.68
80	77.00	9.00	14.00	504.00	78.00	6.00	16.00	491.80	-0.18
100	70.00	10.00	20.00	459.00	72.00	7.00	21.00	450.40	-0.13

Table B.16: 6♡ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	31.00	6.00	63.00	423.50	34.00	4.00	62.00	441.80	0.29
5	54.00	5.00	41.00	774.00	44.00	7.00	49.00	654.00	-1.39
10	60.00	3.00	37.00	845.10	66.00	4.00	30.00	952.20	1.09
20	69.00	4.00	27.00	1011.40	71.00	5.00	24.00	1052.00	0.35
35	76.00	3.00	21.00	1097.20	73.00	4.00	23.00	1068.90	-0.29
50	69.00	4.00	27.00	1004.40	77.00	1.00	22.00	1085.50	0.85
80	78.00	3.00	19.00	1137.10	73.00	2.00	25.00	1035.30	-1.15
100	77.00	4.00	19.00	1134.80	76.00	2.00	22.00	1081.50	-0.68

Table B.17: 6NT results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	36.00	8.00	56.00	495.00	23.00	6.00	71.00	243.30	-2.60
5	55.00	3.00	42.00	756.40	49.00	4.00	47.00	679.40	-0.84
10	61.00	5.00	34.00	903.50	70.00	0.00	30.00	954.40	0.57
20	72.00	4.00	24.00	1042.20	73.00	6.00	21.00	1112.10	0.69
35	71.00	5.00	23.00	1071.00	73.00	6.00	22.00	1091.00	0.26
50	73.00	6.00	21.00	1101.40	77.00	5.00	18.00	1159.60	0.73
80	75.00	6.00	19.00	1132.20	77.00	3.00	20.00	1117.80	-0.19
100	76.00	5.00	19.00	1134.50	73.00	6.00	21.00	1103.40	-0.49

Table B.18: 7♡ results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	26.00	0.00	74.00	437.60	21.00	0.00	79.00	321.10	-0.92
5	41.00	0.00	59.00	807.10	38.00	0.00	62.00	726.80	-0.73
10	55.00	0.00	45.00	1145.50	55.00	0.00	45.00	1136.50	-0.16
20	59.00	0.00	41.00	1238.90	63.00	0.00	37.00	1331.30	0.80
35	62.00	0.00	38.00	1309.20	73.00	0.00	27.00	1564.30	2.04
50	72.00	0.00	28.00	1551.20	68.00	0.00	32.00	1458.80	-0.78
80	76.00	0.00	24.00	1638.60	72.00	0.00	28.00	1546.20	-0.69
100	73.00	0.00	27.00	1566.30	78.00	0.00	22.00	1684.80	1.01

Table B.19: 7NT results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	36.00	0.00	64.00	656.20	33.00	0.00	67.00	547.60	-1.04
5	56.00	0.00	44.00	1151.20	61.00	0.00	39.00	1274.20	1.23
10	66.00	0.00	34.00	1386.20	70.00	0.00	30.00	1503.00	1.11
20	75.00	0.00	25.00	1616.00	74.00	0.00	26.00	1580.80	-0.30
35	80.00	0.00	20.00	1738.00	73.00	0.00	27.00	1565.60	-1.36
50	79.00	0.00	21.00	1693.80	79.00	0.00	21.00	1708.80	0.07
80	80.00	0.00	20.00	1721.00	82.00	0.00	18.00	1777.40	0.50
100	79.00	0.00	21.00	1707.80	77.00	0.00	23.00	1660.40	-0.38

Table B.20: Aggregated Results

No of Samples	No Relocation				Relocation				IMP Difference
	Optimal	Success	Failure	Avg Score	Optimal	Success	Failure	Avg Score	
1	32.07	25.45	42.48	173.53	29.58	27.02	43.39	168.55	-0.02
5	45.03	22.99	31.98	274.97	47.01	22.25	30.74	266.68	-0.02
10	53.47	20.92	25.62	313.84	55.69	22.01	22.30	345.69	0.50
20	61.36	20.13	18.51	371.14	64.27	17.53	18.21	370.88	0.01
35	67.43	17.83	14.74	398.20	65.47	18.51	16.02	389.79	-0.15
50	66.73	18.30	14.97	395.90	69.06	16.92	14.02	406.70	0.17
80	70.27	16.89	12.83	414.33	69.71	15.63	14.67	400.29	-0.19
100	70.05	15.63	14.32	399.28	69.03	15.80	15.17	393.88	-0.12

Appendix C

COMPARISON TABLES BETWEEN THE TWO DEFENDERS

Table C.1: 1♡ Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	54.4	29.8	0.70	54.0	58.2	-0.13	62.3	57.7	0.13
5	65.0	62.4	0.09	47.3	61.6	-0.38	68.8	77.4	-0.26
10	79.2	68.8	0.25	91.9	78.4	0.36	76.8	78.5	-0.05
20	79.6	74.0	0.12	85.1	87.7	-0.07	99.5	97.0	0.06
35	92.8	93.7	-0.03	93.5	95.9	-0.05	95.5	94.6	0.03
50	92.5	90.0	0.07	97.1	93.8	0.06	100.7	94.9	0.18
80	95.5	96.6	-0.02	103.9	97.9	0.17	105.5	98.3	0.21
100	90.6	85.6	0.13	99.9	105.2	-0.15	104.5	97.9	0.19

Table C.2: 1NT Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	17.7	2.2	0.36	14.0	27.6	-0.37	2.6	14.1	-0.28
5	56.7	70.9	-0.41	63.5	60.3	0.06	73.8	71.8	0.04
10	67.0	82.4	-0.43	86.1	76.3	0.18	76.6	81.6	-0.11
20	93.0	78.5	0.37	80.7	93.6	-0.34	87.4	90.2	-0.10
35	97.0	103.2	-0.15	100.4	100.1	0.00	100.7	101.0	-0.03
50	98.0	101.2	-0.08	100.2	94.5	0.13	103.3	98.6	0.13
80	102.3	92.6	0.26	100.1	103.1	-0.05	106.4	101.6	0.16
100	101.7	98.2	0.10	109.2	100.7	0.23	113.7	103.4	0.29

Table C.3: 3NT Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	262.4	200.9	1.01	216.7	248.5	-0.35	279.5	240.2	0.63
5	383.1	375.7	0.08	429.6	425.0	0.03	404.4	360.4	0.77
10	417.8	409.9	0.15	449.4	433.5	0.21	497.4	472.3	0.47
20	484.0	455.2	0.61	488.3	462.9	0.47	489.5	499.3	-0.15
35	556.1	498.3	1.02	517.5	534.0	-0.22	519.1	513.2	0.14
50	541.8	522.6	0.33	522.0	532.0	-0.11	552.9	518.5	0.60
80	545.0	504.5	0.75	522.4	535.8	-0.25	544.8	537.1	0.15
100	521.1	526.9	-0.07	542.1	519.9	0.38	537.6	503.9	0.59

Table C.4: 4♡ Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	233.8	130.4	1.87	236.4	233.1	0.12	187.5	227.3	-0.78
5	307.1	301.7	0.09	298.3	318.1	-0.36	339.7	352.7	-0.28
10	330.5	353.5	-0.33	396.5	397.2	0.13	417.5	452.1	-0.54
20	449.4	430.2	0.39	403.5	499.7	-1.57	465.1	445.4	0.34
35	487.9	479.0	0.12	538.6	506.3	0.57	498.3	492.9	0.12
50	510.5	461.4	0.93	450.6	517.8	-1.10	540.6	537.7	0.05
80	481.3	465.0	0.34	546.8	531.8	0.24	563.7	507.9	0.97
100	557.0	486.1	1.21	540.8	497.4	0.77	564.7	506.8	0.99

Table C.5: 5D Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	133.8	5.8	2.34	150.2	127.4	0.33	179.6	121.4	0.97
5	310.6	251.4	1.00	259.4	317.0	-1.00	292.8	275.2	0.34
10	336.2	374.2	-0.62	384.0	336.6	0.77	399.8	360.6	0.59
20	440.2	378.4	1.04	434.8	436.8	-0.09	457.0	465.6	-0.18
35	462.0	448.0	0.27	459.4	467.0	-0.11	489.8	463.0	0.48
50	454.4	468.0	-0.28	511.4	460.8	0.85	486.2	502.6	-0.27
80	510.4	440.8	1.16	493.4	504.0	-0.15	488.2	491.8	-0.10
100	505.6	462.4	0.70	504.4	459.0	0.75	534.0	450.4	1.41

Table C.6: 6♡ Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	450.8	121.3	3.89	383.6	423.5	-0.63	417.2	441.8	-0.15
5	636.6	679.8	-0.58	706.1	774.0	-0.74	667.2	654.0	0.08
10	700.1	970.5	-2.90	906.3	845.1	0.75	845.1	952.2	-1.10
20	913.3	971.5	-0.56	927.2	1011.4	-1.00	964.1	1052.0	-0.87
35	1101.2	1036.3	0.71	1024.3	1097.2	-0.66	1106.8	1068.9	0.41
50	1097.8	1061.5	0.29	1057.9	1004.4	0.61	1029.6	1085.5	-0.59
80	1094.2	1111.5	-0.15	1072.9	1137.1	-0.65	1122.4	1035.3	0.98
100	1057.0	1083.9	-0.22	1072.5	1134.8	-0.71	1108.1	1081.5	0.42

Table C.7: 6NT Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	461.4	204.5	2.73	472.4	495.0	0.01	437.9	243.3	2.04
5	770.7	805.0	-0.42	773.0	756.4	0.27	713.1	679.4	0.34
10	1020.6	944.5	0.80	865.7	903.5	-0.42	1024.0	954.4	0.81
20	994.2	981.6	0.25	1073.5	1042.2	0.35	1093.2	1112.1	-0.21
35	1075.5	1009.7	0.65	1048.1	1071.0	-0.21	1093.9	1091.0	0.05
50	1075.9	983.9	1.11	1186.0	1101.4	0.87	1076.5	1159.6	-0.76
80	1123.1	1020.1	1.23	1152.5	1132.2	0.13	1185.3	1117.8	0.72
100	1064.5	1031.4	0.40	1156.2	1133.5	0.15	1099.7	1103.4	0.08

Table C.8: 7♡ Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	241.8	-22.1	2.90	270.9	437.6	-1.52	284.9	321.1	-0.33
5	774.0	661.5	0.85	536.0	807.1	-2.56	657.5	726.8	-0.62
10	885.4	951.7	-0.49	1027.0	1145.5	-0.99	1045.1	1136.5	-0.73
20	1170.6	1213.8	-0.50	1005.9	1238.9	-1.80	1146.5	1331.3	-1.54
35	1363.4	1334.3	0.27	1292.1	1309.2	-0.11	1385.5	1564.3	-1.42
50	1317.2	1709.9	-3.23	1504.0	1551.2	-0.30	1407.6	1458.8	-0.48
80	1477.9	1612.5	-1.09	1456.8	1638.6	-1.48	1526.1	1546.2	-0.13
100	1453.8	1570.3	-0.98	1476.9	1566.3	-0.59	1550.2	1684.8	-1.09

Table C.9: 7NT Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	459.4	167.4	2.84	541.2	656.2	-0.92	738.8	547.6	2.24
5	1175.4	1218.8	-0.40	1387.0	1151.2	2.22	1185.4	1274.2	-0.77
10	1478.8	1271.2	1.87	1502.0	1386.2	1.37	1426.4	1503.0	-0.62
20	1594.8	1538.4	0.56	1591.8	1616.0	-0.14	1619.0	1580.8	0.29
35	1566.6	1335.8	1.87	1599.8	1738.0	-1.01	1637.2	1565.6	0.63
50	1620.0	1465.8	1.49	1767.2	1693.8	0.65	1767.2	1708.8	0.61
80	1694.6	1708.8	-0.05	1761.2	1721.0	0.66	1835.8	1777.4	0.61
100	1645.2	1646.2	0.01	1761.2	1707.8	0.47	1814.6	1660.4	1.51

Table C.10: Aggregated Results

No of Samples	Monte Carlo			No Relocation			Relocation		
	Initial	Optimized	IMPS	Initial	Optimized	IMPS	Initial	Optimized	IMPS
1	174.84	93.35	1.40	163.69	173.53	-0.13	168.62	168.55	-0.01
5	255.43	250.00	0.08	256.14	274.97	-0.34	270.87	266.68	0.03
10	280.95	303.04	-0.24	328.02	313.84	0.30	335.93	345.69	-0.11
20	352.69	337.71	0.34	340.37	371.14	-0.48	367.74	370.88	0.00
35	396.32	377.60	0.28	398.11	398.20	0.05	395.68	389.79	0.12
50	398.74	382.43	0.31	383.18	395.90	-0.20	411.60	406.70	0.14
80	397.84	381.52	0.33	411.43	414.33	0.00	424.80	400.29	0.43
100	410.05	388.73	0.41	414.20	399.28	0.28	426.75	393.88	0.60

Appendix D

EXECUTION TIME TABLES

Table D.1: 1♡ Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	5.1	5.3	5.0
5	22.6	26.9	27.7
10	41.6	55.9	60.5
20	73.2	130.3	140.1
35	125.4	254.2	318.6
50	168.0	435.4	543.1
80	257.1	866.0	1343.2
100	317.9	1229.9	2128.3

Table D.2: 1NT Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	4.9	5.1	5.0
5	23.4	27.8	28.9
10	44.0	56.4	55.1
20	85.8	121.6	124.5
35	134.8	229.3	247.2
50	188.8	368.4	454.8
80	295.1	697.6	767.3
100	345.0	922.1	1024.4

Table D.3: 3NT Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	3.9	3.9	3.9
5	17.2	21.9	22.1
10	31.3	45.9	52.2
20	57.1	104.5	122.1
35	100.1	215.4	262.3
50	134.8	368.7	482.3
80	205.4	752.5	1004.8
100	259.0	1126.9	1320.7

Table D.4: 4♡ Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	3.8	3.9	4.0
5	17.2	21.2	23.2
10	32.0	45.7	53.5
20	62.5	115.2	142.9
35	93.5	224.8	283.0
50	128.1	361.4	505.1
80	210.7	752.4	1146.8
100	255.2	1150.8	1487.5

Table D.5: 5♢ Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	3.3	3.2	3.4
5	16.2	19.0	21.2
10	27.9	42.6	48.9
20	51.3	96.3	124.5
35	84.6	216.5	302.8
50	114.7	361.2	530.6
80	168.4	766.5	1123.8
100	226.5	1020.9	1630.6

Table D.6: 6♡ Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	3.6	3.5	3.5
5	17.9	21.5	22.5
10	32.5	46.8	53.6
20	57.0	111.9	149.5
35	85.8	232.8	369.8
50	126.3	389.8	656.4
80	209.5	723.7	1407.3
100	255.2	1131.0	2085.5

Table D.7: 6NT Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	2.6	2.4	2.6
5	12.0	17.0	19.3
10	22.3	40.3	46.5
20	43.5	93.7	133.0
35	68.3	191.4	303.9
50	95.2	319.8	542.3
80	146.0	707.6	1271.4
100	178.1	1152.9	2155.3

Table D.8: 7♡ Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	1.8	1.6	1.6
5	7.9	12.7	15.2
10	15.7	31.8	43.2
20	27.5	78.2	1346.8
35	44.5	181.8	1026.2
50	60.8	330.9	854.1
80	93.7	688.8	1679.8
100	113.7	1111.3	2525.6

Table D.9: 7NT Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	1.5	1.5	1.5
5	7.3	12.7	16.3
10	13.8	31.4	41.3
20	25.8	89.2	125.2
35	41.1	212.8	559.8
50	56.1	373.2	775.6
80	90.0	936.1	2682.4
100	103.1	1314.0	3770.4

Table D.10: Aggregated Average Time (seconds)

No of Samples	Monte Carlo	Our Method (no relocation)	Our Method (relocation)
1	4.2	4.3	4.3
5	19.2	23.3	24.5
10	35.2	49.3	55.0
20	64.9	116.1	145.0
35	105.9	231.4	299.4
50	144.3	386.7	521.7
80	225.7	780.7	1166.1
100	278.1	1139.9	1663.8

Bibliography

[1] P. M. Bethe. *The State of Automated Bridge Play.* Technical Report, New York University, 2010.

[2] Martin J. Osborne. *An Introduction to Game Theory.* Print, 2004.

[3] Nolan Bard, Michael Johanson and Michael Bowling. *Asymmetric Abstractions for Adversarial Settings.* Proceedings of the Thirteenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS):501-508, 2014.

[4] Nolan Bard, Deon Nicholas, Csaba Szepesvari and Michael Bowling. *Decision-theoretic Clustering of Strategies.* Proceedings of the Fourteenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS):17-25, 2015.

[5] Farhad Ghassemi. *Signaling Games.* Report, University of British Columbia, 2005.

[6] Claude E. Shannon. *Programming a Computer for Playing Chess.* Philosophical Magazine, Ser.7, Vol. 41, No. 314, 1950.

[7] M. Bowling, N. Burch, M. Johanson, O. Tammelin. *Heads-up limit holdem poker is solved.* Science 347:145149, 2015.

[8] Sylvain Gelly, Levente Kocsis, Marc Schoenauer, Michle Sebag, David Silver, Csaba Szepesvri and Olivier Teytaud. *The grand challenge of computer Go: Monte Carlo tree search and extensions.* Communications of the ACM Volume 55 Issue 3:106-113, 2012.

[9] Ming-sheng Chang. *Building a Fast Double-Dummy Bridge Solver.* Technical Report TR1996-725, New York University, 1996.

[10] Murray S. Cambell and T.A. Marsland. *A Comparison of Minimax Tree Search Algorithms.* Artificial Intelligence, Volume 20, Issue 4:47-367, 1983.

[11] Samuel H. Fuller, John G. Gaschnig and Gillogly. *Analysis of the alpha-beta pruning algorithm.* Dept. of Computer Science Report, Carnegie-Mellon University, 1973.

[12] M. L. Ginsberg *GIB: Imperfect Information in a Computationally Challenging Game.* Journal of Artificial Intelligence Research 14:303-358, 2001.

[13] Bo Haglund and Soren Hein. *Search Algorithms for a Bridge Double Dummy Solver.* DDS library documentation, http://privat.bahnhof.se/wb758135/bridge/Alg-dds_x.pdf

[14] I. Frank, D. Basin *Search in games with incomplete information: A case study using bridge card play.* Artificial Intelligence 100:87-123, 1998.

[15] Asaf Amit and Shaul Markovitch. *Learning to Bid in Bridge.* Machine Learning 63(3):287-327, 2006.

[16] Mike Cappelletti. *100 Bridge Problems: Using Poker Tactics in Contract Bridge.* Print, 2004.