Cooperative Agents in Euchre

Taras Mychaskiw

April 16th, 2015

Abstract

Typically, agents use a set of rules or follow a specific strategy when given options in order to maximize their expected utility. This project explores such a setting in the card game euchre. Euchre gives an interesting setting where two teams of two agents compete in an incomplete information game. A suite of euchre playing agents are given which each follow a specific and quick strategy in order to maximize not only their personal utility, but their team utility.

Contents

1 Introduction

A good heuristic or static evaluation function in games helps an agent make decisions which maximize their utility. This idea works well, however in team settings, maximizing a personal score may not be a good team strategy. In this project, we investigate this setting in the card game euchre, where two teams of two complete to maximize their team score. Some agents show, among those studied, that cooperation is a much more powerful strategy. This idea was defined in the prisoner's dilemma, where two rational agents that maximize their personal score end up both losing. If these agents cooperated instead, their personal scores and "team" scores would be optimal.

Card games remain popular to study, as their offer interesting settings which lead to good understandings and findings. For example, a common thread in many card games is that they are incomplete information games – other players' hands or the deck is not known to the agent. Perhaps the most interesting about card games is that humans are (generally) much better than the best known artificial agent. Famously, poker was solved in [1], however they study a very small version of poker compared to what is normal in tournament play. Not to discredit their results, but just to say that much work needs still be done in order for artificial agents to consistently beat the best human players in many card game settings. Euchre, compared to other card games, is less conventional in that it is also a cooperative game. This project details various artificial agents for euchre in order to discover a strategy that is superior.

1.1 Euchre

Simply put, euchre is a trick taking card game. It plays very similarly to other trick taking games, such as hearts or bridge. The euchre deck only consists of 24 cards, 9 through ace of each suit. Each of the four players is dealt five cards which makes up their hand. Each players' hand is only known to them. The 21st card of the deck is turned face up to be offered as trump. The remaining three cards remain hidden and are not used in the hand (known as the "kitty").

Players then take turns deciding on if they wish to "order up" the offered trump card. If so, the dealer must pick up the offered card, place it in his hand and secretly discard a card. If the card is not ordered up, players take turns having the option to call a trump suit or pass – but players cannot call the same trump suit as what was offered prior. Players can also choose to "go alone" when they call a trump suit, which forbids their partner from playing the hand in the hopes of scoring more points.

Once trump is called, the main portion of the game takes place. The player left of the dealer leads the first trick. Each player adds a card to the trick in turn. The winner of the trick leads the next trick. This play continues for five rounds, until all players are out of cards. Afterwards, teams count how many trick their won, and the team that took at least 3 of the tricks scores points as:

1 point : if called trump and took 3-4 tricks

2 points: if called trump and took all 5 tricks
2 points: if didn't call trump and took 3-5 tricks
4 points: if went alone and took all 5 tricks

This play continues, shifting who deals every hand, until a team scores 10 points. That is euchre in a nutshell – though trump hasn't been explained. The general play follows the above guidelines. There are additional rules with trump and which cards you can play into a trick, which are explained below.

When a player leads a trick, the card they play determines the lead suit of the trick. Other players must play a card of the same suit if possible, otherwise they can play any card from their hand. The player with the highest ranked card in the trick wins it, however those cards which are not the lead suit or trump cannot ever win the trick. For example, if I lead the 9♡ and you followed with A♢, my lowly 9 is still winning the trick because you did not follow suit (assuming ♢ is not trump).

Trump in euchre is peculiar, and leads to much confusion for beginners. For non-trump suits, the rankings of cards is as you would expect, ace > king > queen .... The lowest trump cards beat the highest non-trump cards (easy enough), however the rankings in among trump is not the same as non-trump. Let ♠ be trump for a concrete example. The ordering of trump then goes J♠ > J♣ > A♠ > K♠ > Q♠ > 10♠ > 9♠. The jack is the highest card in the game, followed by the other jack of the same colour – in our example J♠ is the best and J♣ is the second best (called the "right" and "left" bowers respectfully). The left bower is the confusing part. Though it is printed J♣, it becomes a ♠ as long as ♠ is trump. So when ♣ is led and your only ♣ is J♣, you are not forced to play it as it is not a ♣, but a ♠. Likewise, when ♠ is led, you must play J♣ since it is a ♠. This means there are 7 ♠ in the game, 6 of each red suit and only 5 ♣. Similarly when a red suit is called trump, the other red jack becomes the left bower and it becomes on the trump suit rather than it's printed suit.

1.2 Agent Considerations

With the rules of euchre in mind, we can consider what information is known and important to an artificial agent. When it comes times to play a card into a trick, the following information is known:

the rules of euchre
their own hand
the card offered as trump
all of the cards played so far, and by whom
the trump suit
the lead suit of the trick, is any
the overall rankings of each card

This is all the information an agent has access to. Additional information can be inferred, such as when a player doesn't follow the lead suit, it is known that the player cannot possibly have any cards of that suit in their hand. Making the best use of all the information available will lead to a powerful euchre agent. In the next section, we introduce several euchre agents which make different uses of all this available information. Afterwards, we compare these strategies to each other to determine which is best among them. Lastly, we will discuss the strengths and weaknesses of particular strategies and conclude with ideas for future work.

2 Euchre Agents

The overall objective in euchre is to take as many tricks as possible. This ensures the other team does not benefit from them. Generally, this strategy works well, but it ignores the team aspect of the game. In this section, we give several euchre strategies making various use of the information available in the game. All agents know the basic rules of the game, as a useful agent could not be constructed without this knowledge. Additionally, no agent can cheat. When we say an agent chooses to play a card with some property, it is assumed that the card is chosen from the set of legally playable cards. These strategies range from simple to selfish to complex. These strategies aimed to produce a very quick but good decision when playing a card from their hand into a trick.

2.1 Simple Strategies

The first strategies discussed will be very simple, but serve as a very good base for more complex strategies. These agents only make use of the information given directly to them, their hand. The first agent plays a random card from their hand. This agent exists to serve as a basis for comparison. The next two agents are opposites of each other, one always plays the lowest card in their hand while the other always plays the highest card. Since these agents do not use any additional information, they all play selfishly.

The agents High and Low serve as an excellent foundation for more complex agents. In euchre, the overall goal is to maximize the number of tricks your team earns. Playing High will give you the best possible chance of winning a trick, while Low allows you to throw away your worst card if you resign a trick. Playing a "Middle" strategy would not help compared to High and Low as it seems to give you the worst traits of both the other strategies. With Middle, you could have an increased chance of taking a trick, but if you lost the trick, you've lost a decent card which could have been used later to win a different trick. Do to this, no Middle agent was explored.

2.2 More Complex Strategies

The next agents become slightly more complex. These agents make use of the information given to them in the trick. The first one, called HighLow, behaves as follows:

> if I have a card that can win the trick, play High
> otherwise, there's no hope, play Low

This HighLow strategy proves to perform quite well, and makes a lot of sense. If it is possible to take a trick, playing the best card gives the highest chance of actually winning the trick. If there's no hope of winning a trick, playing your worst card saves your better cards in the hopes that they can win later tricks.

A cooperative version of HighLow was created as well, called CoopHighLow. The logic of this agent is as follows:

> if partner has played in the trick and is winning, play Low
> otherwise, play HighLow

This strategy aims to not take tricks from your partner if it can be avoided. If your partner is winning the trick, saving your good cards for later and allowing them to win will generally be a very powerful strategy. However, if your partner is losing the trick, or hasn't played, the semi-aggressive HighLow strategy makes sense to play to try to win tricks or avoid losing better cards.

2.3 Markov Decision Strategy

The next agent uses a simplified Markov decision process to determine it's hand strength compared to the strength of remaining cards. The agent uses the ratio of these values to guess at whether a card in his hand is strong and has a good chance of winning the trick. At a high level, the Markov agent keeps track of all cards that have been seen so it knows what cards have not been seen. A value is assigned to each card in the deck which represents the strength of that card. When it comes time to make a decision, the Markov agent performs:

$$\text{action}(H, D, \tau) = \begin{cases} \text{play High} & : \frac{Value(H)}{Value(U \setminus D)} \geq \tau \\ \text{play Low} & : \frac{Value(H)}{Value(U \setminus D)} < \tau \end{cases}$$

where H is the set of cards in the agent's hand, D is the set of cards that have been seen and U is the set of all cards. The function $Value(\cdot)$ gives the sum of the values of all the cards in the given set. The threshold parameter τ determines the operating point of the agent, as τ grows, the Markov agent becomes more and more aggressive. Essentially, when the Markov agent sees a high ratio value, it seems it's hand as good, so a High card is played. Otherwise, the weak is viewed as weak and a Low card is played.

The specific values for τ and the cards used in our experiments are discussed in the next section.

2.4 Card Counting Strategy

The CardCounting agent tries to use all available information to it's advantage. This agent keeps track of whether or not each player can possibly have a card or not. When a card is seen, it is known that no player can possibly hold that card in their hand, so this information is tracked. For additional information, whenever a player does not follow the lead suit, it is known that the player cannot have any of the lead suit in their hand. This information is also recorded.

In addition to using all available information, this agent employs a different strategy depending on how many cards are in the trick. It tries to use players being out of a suit (or strong in a suit) to the team advantage. At a high level, the CardCounting agent performs the following thought process before making a decision:

Leading the trick

- if partner is out of a suit and has trump, play Low in that suit hoping the partner can play trump
- if partner is “strong” in a suit compared to both other players, play Low in that suit
- otherwise, play High

Second to play

- if partner can possibly win and is “stronger” than the other remaining player, play Low
- if partner doesn’t have the lead suit:
  if trump was led, play HighLow
  otherwise, if partner has trump, play Low
- otherwise, play HighLow

Third to play

- if partner is winning so far but the last player can win the trick, play HighLow
- otherwise, play CoopHighLow

Last to play

- play CoopHighLow

This method introduces complexity in the hopes of intelligent performance. This agent aims to maximize the number of tricks the team wins through trying to let the partner take as many tricks as possible; except when the partner cannot help, where this agent tries to play selfishly for the benefit of the team.

The word “strong” comes up in the above explanation. The strength of a player in a suit is defined as the sum of the ranks of each card that player can possible have of that suit, where a high ranking means a stronger card. Then, $S_a > S_b$ implies that player a has a stronger hand than player b in that suit.

2.5 Hybrid Strategy

The Hybrid strategy combines the logics of the CoopHighLow, Markov and CardCounting agents. Essentially, this agent using the same Markov decision process as the Markov agent, except when the agent's hand is good, the CoopHighLow strategy is played. Other times, the CardCounting strategy is played. This strategy was created to hopefully compensate for the passive play of the CardCounting agent with the aggressive behaviour of the CoopHighLow agent.

2.6 Monte Carlo Agent

An overly simple strategy in euchre (or any card game) would be to simulate all possible games which follow from the current game state. In incomplete information settings such as euchre, this is simply not computationally feasible. Instead of traversing the entire game tree, the MonteCarlo agent randomly traverses the game tree for a set amount of time, keeping track of how many tricks were won in each path. This is similar to the strategy laid out in [2].

To help reduce the search space further, the MonteCarlo agent retains the idea from the CardCounting agent to keep track of which cards every other player can possibly have. When it comes time to make a decision, a random possible hand is assigned to each other player and games are simulated until finished. This process is repeated until some amount of time has passed, then the card that led to the most number of trick wins is played. At the start of a game, this agent will likely perform poorly as the game tree is still enormous, while nearing the end of the game, the game tree shrinks to a point where it is possible to simulate all possible games which will greatly strengthen play.

3 Agent Comparisons

For each table in this section, we compare two of the strategies discussed in the previous section. At first, we look at teams where the two agents follow the same strategy, followed later by teams consisting of two different types of agents. Unless otherwise noted, all the statistics were taken from a simulation of 10,001 games. In these games, it was forced that the dealer call trump to be ♠ for simplicity, as the agents do not have a strategy for calling trump. Each game was played until a team had 10 points (teams can get 11 points if they score 2 points when they are at 9 points). Each agent was also timed when making it's decisions. Be forewarned, each possible competition is given, so there are many many tables to come.

3.1 Simple Strategies

The simple strategies were compared to each other. Tables 1 through 3 show the results of their competition. As to be expected, Low did poorly overall. Low managed to still win some games however, since Low essentially keeps it's highest cards for the end of the game.

When they get a lucky deal, they can manage to scrape together enough tricks to win a hand. Generally though, playing Low is worse than playing Random. On the other hand, playing High proved to be slightly better than playing Random, though not by very much. Also not surprising is that High beats Low, likely due to winning early tricks. Each of the simple agents perform very quickly, each performing their decisions in very little time.

	Random	Low
Games Won	7656	2345
Tricks Won	284003	226482
Total Score	94343	62520
Time Taken (ms)	477+468	495+454

Table 1: Results of Random against Low.

	Random	High
Games Won	4794	5207
Tricks Won	248331	263064
Total Score	79314	82199
Time Taken (ms)	678+676	800+802

Table 2: Results of Random against High.

	Low	High
Games Won	2550	7451
Tricks Won	229260	286025
Total Score	63942	93393
Time Taken (ms)	1042+1038	1359+1372

Table 3: Results of Low against High.

3.2 More Complex Strategies

Tables 4 through 6 show the results for the HighLow agent against the simple strategies. The results show a staggering favour for the HighLow agent, able to win handily against any of the simple strategies.

	HighLow	Random
Games Won	6902	3099
Tricks Won	279444	226191
Total Score	90862	68105
Time Taken (ms)	926+948	456+499

Table 4: Results of HighLow against Random.

Tables 7 through 9 show the results for the CoopHighLow agent against the simple strategies. Following the trend of the HighLow agent, the CoopHighLow proves to be very powerful compared to the simple strategies.

Table 10 shows the results of an important competition. The CoopHighLow and HighLow agents both prove strong agents, but only one of the tries to cooperate with their team mate. The results favour the CoopHighLow agent, who took on average about a point more per game.

	HighLow	Low
Games Won	8498	1503
Tricks Won	286502	208643
Total Score	97902	54705
Time Taken (ms)	1606+1709	958+997

Table 5: Results of HighLow against Low.

	HighLow	High
Games Won	7578	2423
Tricks Won	278480	201590
Total Score	94017	61710
Time Taken (ms)	1742+1894	1106+952

Table 6: Results of HighLow against High.

	CoopHighLow	Random
Games Won	7729	2272
Tricks Won	281984	211086
Total Score	94601	61277
Time Taken (ms)	2228+2355	1029+951

Table 7: Results of CoopHighLow against Random.

3.3 Markov Decision Strategy

Here, we show the results of two different Markov agents. They follow the same strategy, however the values they assign to each card is different. The first agent uses the number of tricks a card wins out of all possible tricks given the current trump suit as the value for each card. This proved to perform strangely, so an additional value base was formed. Markov2 uses a very similar value base; the value of a card is the number of tricks the card can win which derive from the current trick and given the trump suit. This different value base lends itself more closely to the actual values of the cards. For example, an ace that isn't the lead suit or trump cannot win any tricks, so it should be considered a powerful card. When leading a trick, Markov2 behaves identically to Markov. Both Markov agents used a threshold τ of 0.25 due to there being 4 players in the game.

Something worth mentioning, these Markov strategies precomputed all the values of the cards and stored them for quick lookup when needed during the games. They perform much faster this way, but used quite a bit of memory (around 1GB). However, the time taken to precompute the values was only on the order of a second or two, so in real games they can be calculated on the fly quickly enough.

As mentioned, Markov performs strangely. Markov proves worse than Random, but slightly better than both Low and High. This is strange due to the results given in Table 2, showing that High is better than Random. When played against HighLow and CoopHigh-Low, Markov did not do well at all, losing most of the games and averaging relatively low scores.

The Markov2 agent shows more promise than the Markov agent, showing it is actually slightly better than Random and still better than both High and Low. Markov2 also is quite a bit better than the Markov agent.

This high that Markov2 experienced is quickly brought down by the soul crushing High-Low and CoopHighLow agents. Although Markov2 performed much better than Markov

	CoopHighLow	Low
Games Won	9202	799
Tricks Won	285032	183243
Total Score	100696	44827
Time Taken (ms)	1883+1933	859+874

Table 8: Results of CoopHighLow against Low.

	CoopHighLow	High
Games Won	7874	2127
Tricks Won	278651	197859
Total Score	95331	59278
Time Taken (ms)	915+906	553+516

Table 9: Results of CoopHighLow against High.

	CoopHighLow	HighLow
Games Won	5611	4390
Tricks Won	261354	248731
Total Score	84074	76743
Time Taken (ms)	964+991	977+873

Table 10: Results of CoopHighLow against HighLow.

against these two agents, Markov2 was still trounced. Rather unfortunate.

3.4 Card Counting Strategy

The CardCounting was modelling off of how I personally play euchre, for the most part. Human players tend to keep track of what suit players hold and play accordingly. The CardCounting, similar to the HighLow agent, performs very well against the simple strategies. The results are promising, as the agent achieves very high average scores in each game.

Surprisingly, the more complex strategy modelled after more human play performs not as well as expected against the relatively simple CoopHighLow agent. CardCounting was able to beat the HighLow agent. However, against the CoopHighLow agent, the CardCounting essentially tied, performing slightly worse but taking more tricks. It is due to this disappointing result that the Hybrid strategy was created, in hopes to beat the CoopHighLow agent.

As shown in the previous subsection, the Markov agents are not very powerful. They are swept aside by the CardCounting agent.

3.5 Hybrid Strategy

how the hybrid works

again, how disappointing

although the hybridization worked much better than cardcounting

since markov2 worked so well, hybrid2

	Markov	Random
Games Won	4631	5370
Tricks Won	258091	257844
Total Score	78564	83222
Time Taken (ms)	749+712	471+483

Table 11: Results of Markov against Random.

	Markov	Low
Games Won	5343	4658
Tricks Won	271165	255480
Total Score	82893	78518
Time Taken (ms)	1297+1313	947+899

Table 12: Results of Markov against Low.

	Markov	High
Games Won	5615	4386
Tricks Won	259661	238594
Total Score	84134	76711
Time Taken (ms)	670+702	493+520

Table 13: Results of Markov against High.

	Markov	HighLow
Games Won	3067	6934
Tricks Won	227647	277003
Total Score	68100	91009
Time Taken (ms)	673+733	818+848

Table 14: Results of Markov against HighLow.

	Markov	CoopHighLow
Games Won	2357	7644
Tricks Won	213619	279671
Total Score	62108	94436
Time Taken (ms)	670+706	857+831

Table 15: Results of Markov against CoopHighLow.

	Markov2	Random
Games Won	5373	4628
Tricks Won	266023	248947
Total Score	83148	78497
Time Taken (ms)	775+802	477+446

Table 16: Results of Markov2 against Random.

	Markov2	Markov
Games Won	6501	3500
Tricks Won	273030	238175
Total Score	88540	71045
Time Taken (ms)	1095+1081	962+1034

Table 17: Results of Markov2 against Markov.

	Markov2	Low
Games Won	6528	3473
Tricks Won	277486	241509
Total Score	88888	71113
Time Taken (ms)	1177+1237	789+773

Table 18: Results of Markov2 against Low.

	Markov2	High
Games Won	6502	3499
Tricks Won	273597	228823
Total Score	88825	70785
Time Taken (ms)	1051+1045	777+829

Table 19: Results of Markov2 against High.

	Markov2	HighLow
Games Won	3999	6002
Tricks Won	246909	269371
Total Score	74230	86241
Time Taken (ms)	989+1086	1268+1323

Table 20: Results of Markov2 against HighLow.

	Markov2	CoopHighLow
Games Won	3215	6786
Tricks Won	232695	273115
Total Score	68630	90180
Time Taken (ms)	998+981	1513+1537

Table 21: Results of Markov2 against CoopHighLow.

	CardCounting	Random
Games Won	7615	2386
Tricks Won	283993	214217
Total Score	94358	62629
Time Taken (ms)	1717+1634	667+643

Table 22: Results of CardCounting against Random.

	CardCounting	Low
Games Won	9050	951
Tricks Won	286240	187960
Total Score	100314	46352
Time Taken (ms)	2817+2904	890+908

Table 23: Results of CardCounting against Low.

	CardCounting	High
Games Won	7868	2133
Tricks Won	280486	196979
Total Score	95434	59665
Time Taken (ms)	1062+1126	469+502

Table 24: Results of CardCounting against High.

	CardCounting	HighLow
Games Won	5519	4482
Tricks Won	260753	248582
Total Score	83466	77259
Time Taken (ms)	1221+1215	869+845

Table 25: Results of CardCounting against HighLow.

	CardCounting	CoopHighLow
Games Won	4931	5070
Tricks Won	254396	253984
Total Score	79835	80750
Time Taken (ms)	1360+1320	1043+1048

Table 26: Results of CardCounting against CoopHighLow.

	CardCounting	Markov
Games Won	7603	2398
Tricks Won	279571	215099
Total Score	93918	62358
Time Taken (ms)	1428+1431	999+937

Table 27: Results of CardCounting against Markov.

	CardCounting	Markov2
Games Won	6627	3374
Tricks Won	273151	236744
Total Score	89326	70192
Time Taken (ms)	1621+1568	933+975

Table 28: Results of CardCounting against Markov2.

	Hybrid	Random
Games Won	7765	2236
Tricks Won	283926	211484
Total Score	94993	61292
Time Taken (ms)	1609+1610	562+613

Table 29: Results of Hybrid against Random.

	Hybrid	Low
Games Won	9191	810
Tricks Won	285601	183914
Total Score	100750	44980
Time Taken (ms)	1645+1511	695+759

Table 30: Results of Hybrid against Low.

	Hybrid	High
Games Won	7885	2116
Tricks Won	279051	196344
Total Score	95597	59033
Time Taken (ms)	1389+1405	613+595

Table 31: Results of Hybrid against High.

	Hybrid	HighLow
Games Won	5651	4350
Tricks Won	261381	247699
Total Score	84221	76735
Time Taken (ms)	1548+1637	1139+1122

Table 32: Results of Hybrid against High-Low.

	Hybrid	CoopHighLow
Games Won	4908	5093
Tricks Won	253650	254010
Total Score	80039	80856
Time Taken (ms)	1665+1720	1184+1214

Table 33: Results of Hybrid against CoopHighLow.

	Hybrid	Markov
Games Won	7626	2375
Tricks Won	279820	213780
Total Score	94273	61797
Time Taken (ms)	1648+1725	1017+1025

Table 34: Results of Hybrid against Markov.

	Hybrid	Markov2
Games Won	6872	3129
Tricks Won	273513	231352
Total Score	90506	68017
Time Taken (ms)	1735+1668	1107+1127

Table 35: Results of Hybrid against Markov2.

	Hybrid	CardCounting
Games Won	6872	3129
Tricks Won	273513	231352
Total Score	90506	68017
Time Taken (ms)	1735+1668	1107+1127

Table 36: Results of Hybrid against CardCounting.

	Hybrid2	Random
Games Won	7764	2237
Tricks Won	282805	211655
Total Score	94729	61478
Time Taken (ms)	1694+1658	636+694

Table 37: Results of Hybrid2 against Random.

	Hybrid2	Low
Games Won	9113	888
Tricks Won	285040	183310
Total Score	100451	45177
Time Taken (ms)	1291+1380	563+536

Table 38: Results of Hybrid2 against Low.

	Hybrid2	High
Games Won	7888	2113
Tricks Won	279751	197474
Total Score	95359	59365
Time Taken (ms)	1386+1384	590+629

Table 39: Results of Hybrid2 against High.

what is this nonsense

	Hybrid2	HighLow
Games Won	5501	4500
Tricks Won	260460	248690
Total Score	83300	77467
Time Taken (ms)	1523+1579	1022+987

Table 40: Results of Hybrid2 against HighLow.

	Hybrid2	CoopHighLow
Games Won	4907	5094
Tricks Won	253938	254392
Total Score	80020	80882
Time Taken (ms)	1583+1654	1102+1192

Table 41: Results of Hybrid2 against CoopHighLow.

	Hybrid2	Markov
Games Won	7595	2406
Tricks Won	278489	213471
Total Score	93890	61727
Time Taken (ms)	1581+1524	892+918

Table 42: Results of Hybrid2 against Markov.

	Hybrid2	Markov2
Games Won	6713	3288
Tricks Won	273888	234572
Total Score	89770	69558
Time Taken (ms)	1639+1587	1010+1056

Table 43: Results of Hybrid2 against Markov2.

better than cardcounting again, worse than hybrid

3.6 Monte Carlo Agent

4 Discussion

4.1 Strengths of Particular Agents

4.2 Weaknesses of Particular Agents

cardcounting fails to check for enemies missing suits

montecarlo sucks at beginning which leads to a bad ending

	Hybrid2	CardCounting
Games Won	5085	4916
Tricks Won	254083	254462
Total Score	80599	80379
Time Taken (ms)	1589+1554	1369+1268

Table 44: Results of Hybrid2 against CardCounting.

	Hybrid2	Hybrid
Games Won	4909	5092
Tricks Won	254848	255172
Total Score	80396	81205
Time Taken (ms)	1546+1497	1488+1445

Table 45: Results of Hybrid2 against Hybrid.

5 Conclusions and Future Work

References

[1] Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit holdem poker is solved. *Science*, 347(6218):145–149, 2015.

[2] Furtak. *Symmetries and Search in Trick-Taking Card Games*. PhD thesis, University of Alberta, 2013.