To appear as a chapter in the book 'Game Pogramming'

TR-97-3
Received 14 Feb 97

Computer Bridge Survey

Ian Frank

Electrotechnical Laboratory
Machine Inference Group
Umezono 1-1-4, Tsukuba
Ibaraki, JAPAN 305

Abstract

This paper gives an overview of research into the game Bridge, surveying the published literature on both the bidding and the card play phases of the game, and also discussing commercial programs. The main conclusion to be drawn is that although expert-level performance has been achieved in the problem of Bridge bidding, the overall standard of computer Bridge play in general is very low. In our conclusions, we briefly suggest why this may be so.

© Ian Frank

Bridge

Ian Frank

January 9, 1997

1 Introduction

The poor performance of computer Bridge programs to date has prompted the former World Champion Zia Mahmood to offer one million pounds to the designers of a computer system capable of defeating him. Although it seems that he may now want to withdraw this bet (perhaps worried by the progress of research in other games like chess and checkers), at the moment it still appears that his money should be completely safe. For example, Brent Manley, Bridge expert and reviewer of Bridge programs for the magazine of the American Contract Bridge League, has this to say about the problems facing would-be millionaires [Manley 94]:

> It is generally accepted that writing a computer Bridge program that plays Bridge well is nearly impossible.

Tom Throop, one of the leading producers of commercial Bridge software, is also far from sanguine about the prospects of computer Bridge [Throop 83]:

> The task of writing a Bridge-playing program that demonstrates a high level of intelligence is actually more difficult than writing a similarly intelligent chess program.

It is difficult to simply apply existing game playing techniques to the game of Bridge because of the *incomplete* nature of the information available to the players: unlike in chess, where the positions of all the pieces are known, in Bridge you can only guess which cards your opponents hold. Below, we describe Bridge in a little more detail and then look at how this problem of incomplete information, as well as other difficulties, have been tackled by those working on the game. §3 and §4 survey the academic literature on the sub-tasks of *computer bidding* and *computer defender/declarer play*, with §5 discussing research that falls outside this categorisation. §6 then summarises the state of commercial Bridge programs and finally §7 draws some conclusions.

2 The Game of Bridge

To provide an adequate background, we present a brief introduction of the game of Bridge. To do the game justice, however, would require many pages, and is beyond the scope of this volume. We therefore simply present a summary of the basic rules, and refer those readers interested in further details to one of the many excellent books on the subject (see the Bibliography for suggestions).

2.1 The Basic Rules

Bridge is a card game played with a deck containing 52 cards, comprised of 4 suits (spades ♠, hearts ♡, diamonds ♢, and clubs ♣) each containing the 13 cards Ace, King, Queen, Jack, 10, ..., 2 (we will sometimes abbreviate the first four of these to A, K, Q, and J). The game begins with the chance move of shuffling the deck, and the cards are then dealt between four players, traditionally named North, South, East and West. These players form two teams: North/South against East/West.

Card play starts when one player lays a card on the table, which all the other players then cover in turn (in a clockwise direction) with a card from their own hand. Each round of four cards is called a *trick*, and the winner of one trick becomes the first person to play a card on the succeeding round. Before this play begins there is a *bidding* phase (or *auction*) during which one team wins a *contract* to make a certain number of tricks. One member of the winning team becomes the *declarer* and the other is the *dummy*. The other team are the *defenders*. The defender to the left of the declarer starts the card play, and the dummy's cards are then laid face-up on the table from where they are played by the declarer, along with his own cards. The basic rules that govern card play are:

- The first player in a trick can freely choose which card to play from all those present in his hand.
- Subsequent players must *follow suit* by playing a card of the same suit as the one that started the trick, if they hold such a card — if they do not (*i.e.*, they are *void* in the suit), they can make a free choice from among the remaining cards in their hand.
- The winner of the trick is the player who plays the highest card (ranked by A > K > Q > J > 10 > ... > 2) of the suit led. The only exception to this when there is a suit declared as the *trump suit*; if any trump cards are played, then the player playing the highest trump card is the winner. Playing a trump when you cannot follow suit is known as *ruffing*.

The automation of Bridge play has been carried out by both academics and by producers of commercial software. The resulting programs typically tackle either one or both of the tasks of bidding and card play, which we consider separately below.

3 Computer Bidders

We begin by examining the first broad category of computer Bridge programs — computer bidders. By the very nature of the bidding process, such programs are all basically rule-based systems. The area where they differ is in their approach to interpreting the bids made by the *other* players in an auction.

3.1 The Rule-based Approach

The first Bridge bidding program to appear in the literature is that of Carley [Carley 62]. This system is essentially intended as a card playing system, but it also incorporates a small bidding component, consisting of just four bidding rules with a total of 13 cases. Carley's own assessment of the bidding [Carley 62, Page 7] is:

> The ability level is about that of a person who has played a dozen or so hands of bridge and has little interest in the game.

This level of bidding was acceptable for Carley's program, as it was primarily designed to cope with card play, and also because 'an extremely sophisticated bidding program' was already 'well under way' at Carley's institution. However, there is no record of this more sophisticated program in the literature, so the next Bridge bidder we find is that of Wasserman [Wasserman 70]. Like Carley's program, this is essentially a rule-based system — the difference being largely in the number of rules involved.

Wasserman divides his rules into a collection of *classes* which handle different types of bid, such as opening bids, responding bids or conventional bids. These classes are then organised by a set of *procedures* into *sequences* which specify the allowable orders in which the bids from each class may be used. Examples of these procedures are those that handle sequences starting with a bid of 1♣, or sequences that start with a 'weak two' bid. Wasserman views these procedures as reducing the search space size of the possible bidding sequences by identifying specific sequences to which attention is restricted.

Wasserman's system uses the standard Bridge bidding practice of assigning any hand a number of *points* based on its high cards, or the length of each suit. The strength of a hand measured by these points then helps to determine an opening bid, and this bid in turn determines a specific procedure. This

procedure is then used for all subsequent bids of the partnership, determining the classes of rules to be used for each bid. However, there is a further significant addition. Before each bid is made, the bidder's hand is 're-evaluated' to take into account the history of the bidding sequence, considering such factors as awarding extra points on the basis of length in partner's bid suit(s) and adding/subtracting points based on shortness/length in his unbid suit(s). Wasserman describes the program's use of these factors as attempting 'to bid in the same manner as a human would, using the same decision-making criteria'. We will return to this issue in the following section.

Wasserman presents test results for his program that show it to achieve 'the level of human experts' in unopposed bidding sequences. These tests are based on bidding problems collected from the magazine *Bridge World*, involving actual deals in which each possible contract is awarded a number of points by the editors of the magazine. Any partnership may then test their skill by bidding the hands together and summing their score. On a set of hands for which the editors of *Bridge World* considered a score above seventy to be outstanding, a score above sixty excellent, and a score above fifty to be above average, Wasserman's program scored 58.2. Further, Wasserman added extra bidding classes to his program to enable different bidding systems to be used (thus implementing the 'Standard American' and the 'Kaplan-Sheinwold' systems as well as the original 'Schenken'), which he also tuned to produce a comparable performance.

In addition to normal partnership bidding Wasserman extended his program to handle competitive bidding by adding classes for bids such as *take out doubles*, *penalty doubles* and *overcalling*. In a 30 hand contest against two human Life Masters, split into two sessions, the program was victorious in the first session, defeated in the second, and narrowly beaten overall. Based on these results, Wasserman estimated that 'the program was slightly more skillful than the average duplicate Bridge player at competitive bidding'.

3.2 Interpreting Other Players' Bids

Wasserman points out that Bridge bidding presents 'many opportunities for the exercise of *judgement* and *imagination*, since no Bridge bidding system specifies action to be taken in all cases'. We saw above that Wasserman's program contains algorithms which deal with such considerations by adjusting the evaluation of any hand depending on the bidding history. We also saw that Wasserman justified this approach by appealing to parallels with human thought processes. Others have argued, however, that the 'human' approach more closely resembles reasoning with *explicit inferences* about the meanings of previous bids by the other players. For example, Stanier presents a discussion of the particular sequence of bids where North opens 1♣ and South responds with 1♠. Assuming the standard 'ACOL' system, South's bid could simply be interpreted as showing at least 6 points and 4 or more cards in the spade suit. However, Stanier also presents the following deeper analysis [Stanier 75, Page 376]:

> If we assume that South is making what he believes to be his best bid, then the bid of 1♠ describes his hand more precisely than any other. From this we may infer that he holds more spades than, say diamonds, as with longer diamonds, or with an equal number, he would have bid 1♢ (it being a cheaper bid). Similarly, he cannot hold more than three clubs, or he would have bid 2♣. More information is available from the facts that he did not bid 1♡, 1No Trumps, etc.

The implication is that directly representing such inferences would reflect more closely the type of reasoning carried out by human players of the game. Also, the ability to make inferences in this form would be highly useful to a card playing component called upon to make or defend a contract. However, the performance of programs attempting to explicitly represent such information has yet to be proven in practice. For example, [MacLeod 91] describes a system which attempts to use the bids of any player to build a 'picture' of the cards he holds. Unfortunately, the representation used for capturing such information is not capable of coping with disjunction, and when any bid with more than one possible interpretation is made, information is therefore lost. This illustrates an important obstacle in the implementation of such systems: designing a representation language in which expressions describing

all the possible inferences about each hand can be automatically generated. One technique for this which could be borrowed from mathematical reasoning is the use of a set of confluent rewrite rules to produce expressions in a normal form. However, we can already see from the above discussion that there are many possible types of inference that can be made about a single hand, including the number of cards in each suit, the maximum and minimum number of total points, the maximum and minimum length in each suit, and the relative lengths of each suit. Further, the position of particular high cards is often important, as is the breakdown of a hand's overall points into the points held in each suit. Since most of these possible types of inference can interact, it becomes difficult to prove termination of any given re-write set. Such problems were encountered in the work of [Asher 93] and [Green 95].

3.3 Incorporating Look-ahead

An attempt to create a more general system capable of Bridge bidding is found in [Stanier 75]. Stanier's aim was to 'keep knowledge of Bridge peripheral to the program, and use a general core'. Stanier reasons that whilst bidding systems 'provide useful guidelines for the bidding, they by no means make the process of bidding mechanical', and he therefore argues that an element of planning should occur during the bidding phase in order to produce realistic performance. Although this is a worthy aim, the problems involved are even larger than those in making inferences from single bids, and Stanier eventually concedes that 'it was found necessary to put this [domain] knowledge in the core itself'.

Another system which proposes the use of look-ahead search over the space of possible bids is given in [Gambäck *et al* 91]. In this system, the rule-based portion is used simply to *suggest* possible bids for a player. These bids are then evaluated by conducting a look-ahead search; at each node, the valid inferences about each hand are made, a set of hands consistent with the constraints is generated, and a neural network is used to assess the chance of making the contract under each deal. The top-level control of this process is carried out by the further addition of a Bayesian planning system. Unfortunately, no test results are given.

3.4 COBRA

Judging purely by performance, the best bidding systems are those which do no lookahead search and deal with the problem of inferencing using quantitative adjustments to hand evaluations rather than by explicitly representing inferences. The epitome of this approach is Lindelöf's COBRA [Lindelöf 83].

In COBRA, each player's hand is assigned a number of points as a 'measure of the probable trick winning power of a hand'. At the beginning of the game, with no information available about the other hands, this evaluation is referred to as the *basic points count* (BPC) and is based on the assignment of points for each high card (A=4, K=3, Q=2, J=1), with extra points added and subtracted for considerations such as distribution, controls, shape and unprotected honours. The BPC 'is chosen so that any opening promises 13+ points' (except for bids by the third player). Its exact specification was produced by examining 'several million hands' which were randomly generated and assessed largely automatically.

During the auction, the points count of any player is adjusted and is then referred to as the *distributional points count* (DPC). The actual rules for this are not complicated, but take three pages of Lindelöf's book to describe [Lindelöf 83, Pages 31–33]. Basically, points are added or subtracted from the BPC for short suits, fit with partner, fit with the opponents, and other such considerations. These modifications are designed so that a joint value of 28 between two hands 'gives the partnership a better than 50% chance of making ten tricks in their best trump suit' with any additional trick requiring another 3 points. Again, these modification rules were tested by generating large numbers of random hands.

Lindelöf claims that COBRA's bidding is of world expert standard, and provides evidence that in fact make this assessment sound rather modest. On a collection of 37 sets of 'challenge' hands from Bridge Magazine, taken from the period 1973 to 1979, COBRA scored an average of 89% and only once finished below the leading human pair. On another test, COBRA was used to replace the bidding of the human teams in the 1975 Bermuda Bowl final. Lindelöf claims that COBRA would have improved the North American score by 98 IMPs (International Match Points) or the Italian score by 69 IMPs — enough to translate the narrow Italian victory of 214–189 into a comfortable win either way.

An independent test of COBRA was carried out by David Grabiner [Grabiner 94]. Under similar criteria to that of Lindelöf, Grabiner used COBRA to replace the human players in the 1987 Bermuda Bowl and Venice Trophy. The results of this test are summarised in Figure 1, showing that the total swing produced by using COBRA would have been +74 over the 512 deals. Although this is still a positive result, it is much less than Lindelöf's +167 in 192 boards. However, Grabiner attributes some of this discrepancy to the likelihood that Lindelöf's book is not complete in some areas, and is also open to mis-interpretation.

Tournament	Actual Result	Difference when replacing one team with COBRA	
Bermuda Bowl	US – GB 241 – 148	COBRA – GB +15	US – COBRA +43
Venice Trophy	US II – France 254 – 219	COBRA – France -2	US – COBRA +18

Figure 1: Grabiner's test results for COBRA

Whatever the exact strength of COBRA it seems indisputable that it makes expert-level Bridge bidding a reality. The approach has been criticised, however, by [Gambäck *et al* 91] on the grounds that it will 'only have a very limited ability to reason about its opponents bids' and that this will be a 'serious problem', since 'successful play requires that the bidding component is able to pass on the (probabilistic) information extracted from the bidding to the playing component'. This objection can be tackled by noting that a separate deduction system, such as that of [Quinlan 79] (see §5) could be used to produce these types of inferences. Also, it should be clarified that it is not the case that COBRA is only a world-class bidder 'when it is allowed to bid undisturbed towards a slam' as [Gambäck *et al* 91] suggests. In fact, the test results summarised above were all produced under conditions of competitive bidding by all four players. COBRA has a full set of rules to cope with both defensive and pre-emptive bidding, and indeed the random hands on which COBRA was optimised necessarily allowed for bids by all four players, since a bidding system designed for uncontested auctions would have been heavily skewed towards slow development. The experimental evidence, then, is that the technique of calculating a numerical measure of each hand simplifies the intricacies of reasoning with qualitative information very effectively. One of the few problems overlooked by such an approach is the desirability of reaching the optimal contract whilst giving as little information about your own cards to the opposition. However, this may also be incorporated in future versions of the system.

4 Computer Defender/Declarers

The search space in a Bridge card play problem contains at least 9.91×10^{22} legal play sequences [Frank 96]. The main pre-occupation of programmers attempting to automate card play has therefore been the effective control of the search through this space of possibilities. Below, we identify and discuss four techniques that have been used for this purpose.

4.1 Rule-based Systems

We have already mentioned the early research of [Carley 62] in the context of bidding. In fact, the rule-based operation of Carley's bidding system is almost identical in principle to that of the play program with which it is coupled; the issue of search is side-stepped completely by simply specifying a set of rules

for actions to take in certain situations. Carley describes the resulting system as 'essentially a long list of rules of thumb'. An example of one of these is:

> Defender, can follow suit, third card to trick, second card high, probability the trick can be won in this hand is more than 0.6 $\longrightarrow$ play lowest such winner

As might be expected from a collection of such rules, the performance is not particularly robust, although Carley claims it plays 'quite respectably'. However, it is interesting to note that this approach has survived until very recently, and indeed was the method adopted by the Bridge King program which won the gold medal for Bridge at the 1992 Computer Games Olympiad. There is little description in print about this system, but some comments by Hans Leber — one of its designers — are reported in the 19 September 1992 edition of New Scientist magazine [Geake 92]. He describes the program a little, revealing that it does no look-ahead search at all — a design decision he justifies by claiming that its decisions must be made in 'seconds rather than minutes'. Apparently, 'Bridge King simply deduces as much as it can about where the cards are, then follows a set of rules'. Its level is indicated by the claim that it can 'beat a hobby player, but not a tournament player'. 'Our program still makes silly mistakes,' says Scholz, 'but we feel happy because others make bigger mistakes'.

An interesting attempt to tune the rules in such a system is presented in [Napjus 69]. His approach was to use a machine-learning technique that would 'discover' the best options in any given situation by playing out rival possibilities against a large number of different subdivisions of the outstanding cards. Each possible *line of play* was given a weight and the one which performed better on most of the subdivisions had its weight strengthened slightly. The problem with this approach is that it suffers from a kind of 'Catch-22' situation: in order to test the lines of play open to the declarer, an algorithm for generating defensive plays is necessary, but defenders judge the selection of their defensive play by reasoning about which lines of play the declarer is likely to be following. In practice, Napjus' system could be easily misled by defence that was very good into thinking that a good line of play was in fact inferior. The learning procedure eventually relied on an algorithm for generating defensive plays that was 'good, but not too good'.

4.2 Tactics

Given that the technique of simply using rules has obvious limitations, we now consider how search can be incorporated into card play programs.

The obvious problem when using look-ahead in Bridge is the size of the search space involved. However, one way to avoid examining every possible play sequence in this space is to limit consideration at each stage of planning to a pre-determined set of options. This was the approach taken by Stanier, who constructed plans by linking together *schemas* corresponding to types of plays typically used by human players [Stanier 76]. Stanier's system recognises six schemas: taking top tricks, drawing trumps, ruffing, establishing long cards, finessing, and the promotion of non-masters. However, the exact nature of these schema is never clearly defined and no performance tests or evidence of a working implementation are given.

A more successful attempt at constructing a planner with restricted options is the FINESSE system [Frank 96, Frank 91]. This system restricts itself to considering plays in a single suit, identifying seven types of possible play, or *tactics*. The applicability conditions of each of these tactics are specified as *methods*, which the planner then uses to solve any given problem by generating the entire tree of legal combinations of tactics. Using this technique, a probability of success can be assessed for each option, and later versions of the system also incorporated the ability to generate textual explanations. A typical example of such an explanation [Frank *et al* 92, Page 75] is:

> This leads to 2 tricks if West holds at least one of the King or Queen and West holds the ten, or if West holds both the King or Queen, East holds the ten, and East holds the remaining four low cards.

This approach of specifying higher level operators is also employed in the ASPEN system [Nygate & Sterling 93], which in fact even uses the same terminology of 'tactics'. However, ASPEN is less able to distinguish between actual probabilities than FINESSE, and simply attaches one of the three designations `certain`, `usual`, or `possible` to the outcome of any sequence of actions. The general framework of tactics and methods also fits the TIGNUM system developed by [Smith & Nau 93], who explain that 'to represent the tactical and strategic schemes of card-playing in Bridge we use instances of multi-agent methods'. However, they characterise their system as an example of forward-pruning — a search algorithm which picks *one* choice at any point in the search [Smith & Nau 94].

An extreme application of the use of tactics can be found in the work of [Khemani 94]. For Khemani, 'the main idea is to retrieve complete plans from memory, which contains the cumulative experience internalised into easily accessible structures' [Khemani 94, Page 287]. These high-level objects, which Khemani calls 'Thematic Actions', are clearly the same notion as FINESSE's methods. However Khemani's TA's typically represent longer strings of actions: play sequences composed of four or five tricks, rather than the single tricks of FINESSE.

In fact, Khemani's system has no method of comparing the worth of competing options, so it becomes highly important to ensure that 'best' matching TA is always returned first for any given problem. However, since this approach necessarily entails constructing a large database of possible situations, it suffers from the obvious drawbacks of ensuring completeness (*i.e.,* that a reasonable play sequence will be selected for every card combination) and of maintaining the integrity of the database to ensure the 'best' play for any situation is always the first returned. Further, even if such an approach could eventually be made to play the game of Bridge perfectly it would represent a rather unsatisfactory solution in terms of explaining its actions. Consider, for example, the distinction made by Victor Allis [Allis *et al* 91] between the *cracking* of a game (achieved by producing a computer program which can play perfectly) and the *solving* of a game (which further requires that the algorithm embodied by the program be explicable in human terms). Khemani himself admits that 'the knowledge contained in the TA cannot provide explanations as to why this play is best'.

The lack of a method for evaluating the worth of plans seems quite a severe limitation, since there will often be more than one way to make a contract, and in such situations some measure of the plan which is most likely to succeed could prove very important. Further, the object of Bridge declarer play is typically not simply to make a contract, but also to make some overtricks (especially in situations like match-pointed pairs, where the same cards are played by many players and the results compared). Khemani's control system of accepting the first plan which has a chance of making the contract therefore seems overly simplistic and indeed, the plans produced by Khemani's system tend to demonstrate this weaknesses. For example, in the deal of Figure 2, the object is for the declarer (South) to win all 13 tricks with no trumps. Khemani's system counts the *top tricks* (the North and South hands have 11 between them: three top spades, one top heart, four top diamonds, and three top clubs) and then tries to identify suits in which extra tricks could be made.

Diamonds obviously provide an extra trick unless the cards are split 5–0 between the defenders, and the spades produce an extra trick if they are split 3–2. Khemani's system realises this and plans to play for an extra trick in both suits, starting with the spades. After cashing the A♠ and K♠, however, it discovers the spade break, and then re-plans, eventually deciding to re-enter the South hand by winning with the Q♠, and then finessing the Q♡ (this play involves leading the 7♡ in the hope that West holds the K♡: if West plays the King, the declarer obviously wins with both the Ace and Queen; if West plays low the declarer wins with the Queen by playing it immediately, and then wins with the Ace on the next trick). Although this new plan results in making the contract, the more natural way of representing this approach would be to explicitly state that the plan being considered is to make extra tricks in the spades and diamond suits, with the heart finesse as a backup source.

Readers with some Bridge knowledge may also notice that in fact there are ways for declarer to further increase his chances in Khemani's problem. If just *one* of the diamond or spade suits fails to break favourably, an extra trick can be produced without resorting to the finesse if the declarer sets up a *simple squeeze*. For example, in Figure 2 the diamonds break but the spades do not. By playing all the winners in spades and diamonds as well as the K♣ and Q♣ declarer can bring the play to the situation

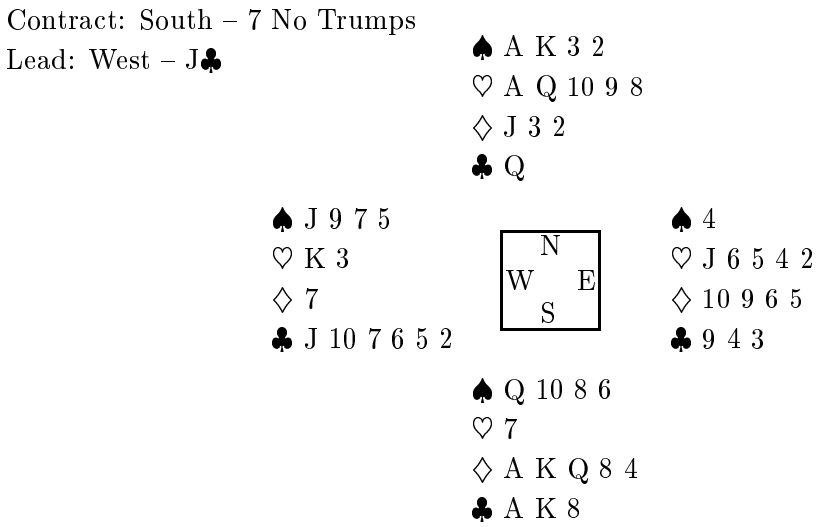

Figure 2: An example Bridge deal from [Khemani 94, Page 292]

of Figure 3. When South now cashes the A♣, West cannot help allowing the declarer to win the final two tricks. If he plays the J♠, declarer will win with the 10♠ and then the A♡. If he plays the 3♡, declarer will win with the A♡ and Q♡. This play is called a squeeze because West is literally squeezed into throwing away a card that he needs to keep. Note that the play still succeeds even if East and West's cards are swapped in Figure 3. We can therefore win whenever the player with the long spades also holds the K♡. A similar squeeze situation can also be set up if it is the diamonds that break unfavourably instead of the spades, and also even if *both* the diamonds and spades break unfavourably, if we can finesse the Q♡.

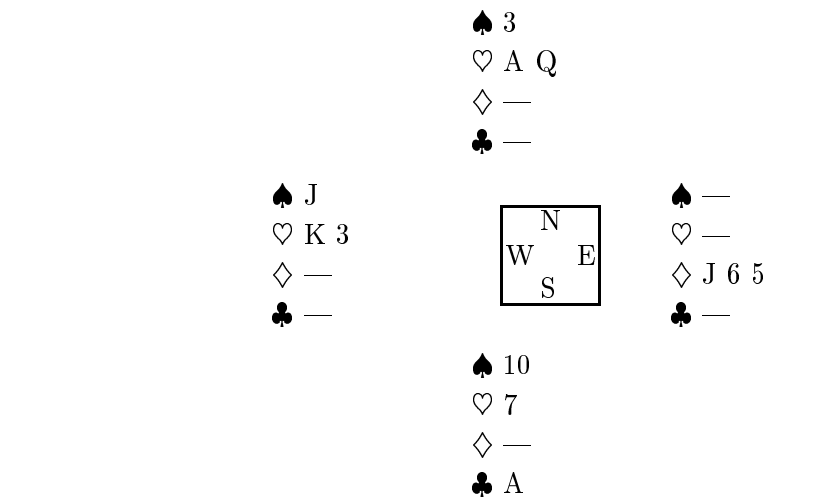

Figure 3: Setting up a squeeze in Khemani's example

Such squeeze plays are very subtle, and currently beyond the means of any card play program (although see the discussion of PYTHON in §5). This example therefore illustrates the points that Bridge situations are in general difficult to solve (often requiring lookahead to the very end of a contract), and also that it is not uncommon to find errors in deals presented in Bridge papers.

4.3 Tackling Card Combinations

A third way to reduce the search space size in Bridge is to consider only the sub-problems of the *card combinations* in individual suits. This was originally suggested in the domain of Bridge by [Berlin 85]. In Berlin's case, the simplification was made in order to demonstrate a modification to the classical planning framework for deciding how to order possibly interacting subgoals. It is based, however, on the common problem-solving strategy of identifying sub-problems which are solved independently.

The FINESSE system described above benefited from the reduced search space in individual suits (coupled with the use of tactics) to the extent that it could generate the entire tree of legal play sequences for any given card combination. Also, [Kibler & Schwamb 92] make use of the reduction in search space offered by this approach to implement an algorithm for deciding the best moves in single-suit problems. However, this system does not employ tactics, and thus searches through a much larger space (typically in the order of 10^4 compared with the 10^2 of FINESSE).

The problem with tackling card combinations independently, of course, is that they typically interact. Thus, even if sub-plans can be generated for each suit, combining them together into a global plan may not be easy. The dependencies which may exist between the suits give rise to such considerations as *entries*, *unblocking*, and *end-plays* as well as the squeeze plays we examined above. However, human players seem to learn techniques for reasoning about such complications with relative speed. [Gambäck *et al* 91] put forward the suggestion that such problems can be tackled by reasoning about properties such as *resources*. These ideas are not developed beyond a very superficial level, however, and are not implemented. Two systems that have attempted to implement this approach are those of [Nygate & Sterling 93] and [Frank 96], both mentioned already. Nygate's system contains a tailor-made 'strategy synthesis' module, whereas Frank's system develops a general-purpose planning algorithm that uses probabilistic reasoning to assign resource producers to consumers.

4.4 Double-dummy Programs

Some authors have tried to simplify the task of automating Bridge card play by solving the easier situation where all the players reveal their cards to each other [Berlekamp 63, Wheen 89]. In this circumstance, the uncertainty over incomplete information is completely removed, and thus the search space of possible moves is also reduced. Since this complete knowledge situation is akin to the opponents placing their cards on the table in the same manner as dummy, this scenario is often described as *double-dummy* Bridge.

The relative ease of solving double-dummy problems has prompted some researchers, for example Levy [Levy 89] and Ginsberg [Ginsberg 96], to suggest Bridge-playing architectures that work by examining a statistically significant number of the possible distributions of the outstanding cards that are consistent with a player's knowledge. They speculate that in any given situation the use of search reduction techniques (such as alpha-beta pruning) would enable the minimax value of each possible action to be established in each of these randomly generated sub-problems, and the best overall action to be predicted by suitably combining these values. However, it is known that, even in very simple incomplete information games, algorithms like those suggested by Levy and Ginsberg will yield suboptimal results against 'best defence'. In particular, [Frank 96] formalises the problems of *non-locality* and *strategy fusion* which will afflict such algorithms independently of how many possible distributions are considered. The extent to which these shortcomings degrade actual performance, however, remains an open research issue.

5 Miscellaneous Bridge Research

There is some Bridge research which does not fit the computer bidder/declarer/defender classification. Nygate, for example, has produced an expert system called PYTHON for identifying squeeze situations [Nygate 84, Nygate & Sterling 90]. As we have already seen, squeezes are an advanced strategy which many human players have difficulty understanding, so even automating their recognition is an achievement. Nygate himself, however, recognises that PYTHON has some important limitations. Firstly, squeeze play is an end-game strategy that normally arises when there are 3 to 7 cards remaining. PYTHON is

only capable of recognising such squeeze situations when they are immediately applicable, producing a plan which creates an extra trick whenever the cards are positioned favourably. Although it was later coupled to a primitive Bridge playing program which could execute these plans [Lustig 85], the system is *not* capable of suggesting what order the cards should be played in to *reach* a potential squeeze situation. Secondly, when the possibility for more than one squeeze play exists, PYTHON cannot choose which one is more likely. This is because it 'has no knowledge of the distribution of the outstanding cards, in contrast to a Bridge player who can infer a great deal about the adversaries cards' by clues drawn from the bidding and the play [Nygate 84, Page 45].

Another unusual system is that of [Quinlan 79] — a knowledge-based system capable of making conclusions about card location based on information from the bidding and the opening leads. This system can either locate honours with one of the defenders with certainty, or it can make less specific assertions of the form 'x holds either y or z'. It is not capable of making probabilistic statements, but even so the inferences are of a type that would be useful to a card playing program. Actually producing a system capable of explicitly utilising such inferences would undoubtedly represent a step forward in the state of the art of Bridge programs.

6 Commercial Bridge Programs

[Throop 83] traces the development of Bridge-playing programs (mostly those resulting in a consumer product, but also some of a research nature) as far back as 1957. Many of the systems Throop describes achieve a reasonable level of performance, especially in bidding. Unfortunately, however, the descriptions of the algorithms for card play are far from enlightening. For instance, Throop's complete description of one of his own programs is as follows:

> Behaviour of the program is heuristic in nature. The program evaluates trick objectives, formulates goals, and executes the play of the declarer's and the dummy's cards. The program performs in accordance with heuristic principles I have always practised in tournament play and which I incorporated into the computer program.

Throop makes no attempt to detail the 'heuristics' he talks of, leaving readers to imagine for themselves the kinds of algorithms involved. The overall impression is therefore of programs that rely on some kind of pattern-matching to identify common game states, and *ad hoc* modifications to cope with unusual situations.

Since Throop's book was published, many new commercial programs have been developed. In general, whilst some of these programs may be reasonably good bidders, there are as yet no programs which can play the cards with any degree of expertise. This point is emphasised by, for example, the recent review of computer software in the magazine of the American Contract Bridge League [Manley 94], which found that

> While most of the programs received good to excellent ratings on ease of use, graphics, speed and features, scores dropped dramatically when testers rated the skill of the programs - that is, expertise in dummy play and defence. Only three of the nine programs rated above average in skill.

7 Conclusions

We have looked at literature on the game of Bridge, examining both academic research and commercial products. The important conclusion is that although expert level performance has been achieved in the sub-problem of Bridge bidding, the overall standard of computer Bridge play in general is very low. This contrasts sharply to the success of computers in other games. In chess and checkers, for example, computer programs are amongst the top players in the world. In Bridge, however, a history of academic research and a proliferation of more than two dozen commercial software packages has failed to produce a system capable of competing with even good novice human players.

The poor performance of Bridge programs may be partly due to the incomplete information of the game, but it may also reflect the extraordinary ability of humans to play the game. In Checkers, with a comparable search space size to a Bridge declarer play problem, no player (human or otherwise) can sit down at the table at the beginning of a game and announce that they have decided upon their best moves through to the end. However, this is in fact what human declarers do in Bridge, forming and comparing complete strategies in a matter of tens of seconds before playing to trick one. How can this ability be explained? One possibility is that human declarers employ the common problem-solving strategy of splitting an initially complex global problem into several smaller sub-problems which are then worked on independently. In games like chess and checkers, where such decompositions are difficult to identify, both humans and computers achieve limited lookahead (the computer by sheer brute force, humans by recognising promising patterns and areas of the search space), and the gap in performance is now being closed. In Bridge, however, limited lookahead is comprehensively outclassed by a human opponent who can project play through to the crucial end stages of the play by considering how plans for each suit can be pieced together.

Recall the analysis of the team designing the CHINOOK checkers playing program [Schaeffer *et al* 93]. They cite an instance from a match against the World Champion Marion Tinsley, where CHINOOK played its 10th move and on the basis of a 17-ply search claimed a small advantage. However, the game was lost in 35 moves and afterwards Tinsley revealed that 'on move 12 he had seen to the end of the game and knew he was going to win'. The best estimate of the CHINOOK team was that this feat involved 'selective searches of over 60 ply'. In a game such as Bridge with a comparable search space size and human opponents that are regularly able to project the play through to its conclusion, such occurrences are likely to be even more frequent.

References

[ACBL 94] ACBL. *The Official Encyclopedia of Bridge*. American Contract Bridge League, Inc., 2990 Airways Boulevard, Memphis, Tennessee 38116-3875, fifth edition, 1994.

[Allis *et al* 91] L.V. Allis, H.J. van den Herik, and I.S. Herschberg. Which games will survive? In D.N.L. Levy and D.F. Beal, editors, *Heuristic Programming in Artificial Intelligence 2 – The Second Computer Olympiad*, pages 232–243. Ellis Horwood, 1991.

[Asher 93] David Asher. Convent — a bridge bidding system. Unpublished M.Sc. thesis, DAI, 1993.

[Berlekamp 63] E.R. Berlekamp. Program for double-dummy bridge problems — a new strategy for mechanical game playing. *Journal of the ACM*, 10(4):357–364, 1963. Also in D. Levy, editor, *Computer Games II*. Springer Verlag, New York, 1988.

[Berlin 85] Daniel L.S. Berlin. Span: Integrating problem solving tactics. In *Proceedings of IJCAI-85*, pages 1047–1051, Los Angeles, California, 1985.

[Carley 62] Gay Carley. A program to play contract bridge. Unpublished M.Sc. thesis, Dept. of Electrical Engineering, Massachussets Institute of Technology, Cambridge, Massachussets, June 1962.

[Frank 89] A. Frank. Brute force search in games of imperfect information. In D.N.L. Levy and D.F. Beal, editors, *Heuristic Programming in Artificial Intelligence 2 – The Second Computer Olympiad*, pages 204–209. Ellis Horwood, 1989.

[Frank 91] I. Frank. Finesse: An adaption of proof-planning techniques to declarer play in the game of Bridge. Unpublished M.Sc. thesis, DAI, 1991.

[Frank 96] Ian Frank. *Search and Planning under Incomplete Information: A Study using Bridge Card Play.* PhD thesis, Department of Artificial Intelligence, Edinburgh, 1996.

[Frank *et al* 92] I. Frank, D. Basin, and A. Bundy. An adaptation of proof-planning to declarer play in bridge. In *Proceedings of ECAI-92*, pages 72–76, Vienna, Austria, 1992. Longer Version available from Edinburgh as DAI Research Paper No. 575.

[Gambäck *et al* 91] Bjorn Gambäck, Manny Rayner, and Barney Pell. An architecture for a sophisticated mechanical bridge player. In D.N.L. Levy and D.F. Beal, editors, *Heuristic Programming in Artificial Intelligence 2 – The Second Computer Olympiad*, pages 87–107. Ellis Horwood, 1991. Later version also available as Cambridge University Computer Laboratory Tech Report 299, May 1993.

[Geake 92] Elizabeth Geake. Playing to win. *New Scientist*, pages 24–25, 19 September 1992.

[Ginsberg 96] M. Ginsberg. How computers will play bridge. *The Bridge World*, 1996. Also available for anonymous ftp from dt.cirl.uoregon.edu as the file /papers/bridge.ps.

[Goren 86] Charles H. Goren. *Goren's New Bridge Complete.* Century Hutchinson Limited, 1986.

[Grabiner 94] David Grabiner. Personal communication, September 1994.

[Green 95] Stephen Green. Drawing inferences from bridge bidding. 4th Year Project Report, DAI, 1995.

[Khemani 94] Deepak Khemani. Planning with thematic actions. In *Proceedings of the Second International Conference on AI Planning Systems (AIPS-94)*, pages 287–292, Menlo Park, CA, 1994. AAAI Press.

[Kibler & Schwamb 92] D. Kibler and K. Schwamb. Complete contingency planners. Technical Report UCI TR 92–94, University of California, Irvine, 1992.

[Levy 89] D.N.L. Levy. The million pound bridge program. In D.N.L. Levy and D.F. Beal, editors, *Heuristic Programming in Artificial Intelligence – The First Computer Olympiad*, pages 95–103. Ellis Horwood, 1989.

[Lindelöf 83] E.T. Lindelöf. COBRA*: The Computer-Designed Bidding System.* Victor Gollancz Ltd, London, 1983. A volume in the Master Bridge Series (series editor Kelsey, H.W.), ISBN 0-575-02987-0, available from IPBM, 455 Alfreton Road, Nottingham NG7 5LX, England, Phone: +44 602 422615.

[Lopatin 92] A. Lopatin. Two combinatorial problems in programming bridge game. Computer Olympiad, 1992.

[Lustig 85] R. Lustig. Hibrid: A hierarchically designed bridge playing program. Unpublished M.Sc. thesis, Weizmann Inst. of Science, Rehovot, Israel, 1985.

[MacLeod 91] J. MacLeod. Microbridge — a computer developed approach to bidding. In D.N.L. Levy and D.F. Beal, editors, *Heuristic Programming in Artificial Intelligence – The First Computer Olympiad*, pages 81–87. Ellis Horwood, 1991.

[Manley 94] B. Manley. Bridge software: Moving forwards or backwards? *The Bulletin, American Contract Bridge League*, October 1994.

[Napjus 69] C.N. Napjus. *Declarer: A Learning Bridge-Playing Program which Generalizes and Infers.* PhD thesis, University of Washington, 1969.

[Nygate & Sterling 90] Y. Nygate and L. Sterling. Python: An expert squeezer. *The Journal of Logic Programming*, 8(1 and 2), January/March 1990.

[Nygate & Sterling 93] Y. Nygate and L. Sterling. Aspen — designing complex knowledge based systems. In *10th Israeli Symposium on Artificial Intelligence*, pages 51–60, 1993.

[Nygate 84] Y. Nygate. Python: A bridge expert on squeezes. Unpublished M.Sc. thesis, Weizmann Inst. of Science, Rehovot, Israel, 1984.

[Quinlan 79] J.R. Quinlan. A knowledge-based system for locating missing high cards in bridge. In *Proceedings of IJCAI-79*, pages 705–710, Tokyo, 1979.

[Reese & Bird 83] Terence Reese and David Bird. *Bridge, the Modern Game.* Faber and Faber Ltd, 1983.

[Reese & Dormer 91] Terence Reese and T. Dormer. *The Play of the Cards.* Robert Hale Limited, 1991.

[Schaeffer *et al* 93] J. Schaeffer, P. Lu, D. Szafron, and R. Lake. A re-examination of brute-force search. In *Games: Planning and Learning, Papers from the 1993 Fall Symposium*, pages 51–58, AAAI Press, 1993.

[Smith & Nau 93] S.J.J. Smith and D.S. Nau. Strategic planning for imperfect information games. In *Games: Planning and Learning, Papers from the 1993 Fall Symposium*, pages 84–91, AAAI Press, 1993.

[Smith & Nau 94] S.J.J. Smith and D.S. Nau. An analysis of forward pruning. In *Proceedings of AAAI-94*, pages 1386–1391, 1994.

[Stanier 75] A. Stanier. Bribip: A bridge bidding program. In *Proceedings of the 4th IJCAI*, Tbilisi, USSR, 1975.

[Stanier 76] A. Stanier. Planning to make tricks at bridge. In *Proceedings of the AISB Summer Conference*, pages 256–265, 1976. Also in D. Levy, editor, *Computer Games II*, Springer Verlag, New York, 1988.

[Throop 83] T. Throop. *Computer Bridge.* Hayden, 1983.

[Walker 96] Gabrielle Walker. Beaten in spades. *New Scientist*, pages 26–27, 16 November 1996.

[Wasserman 70] A. Wasserman. Realisation of a skillful bridge bidding program. In *Proceedings of the Fall Joint Computer Conference — AFIPS*, Houston, Texas, 1970.

[Wheen 89] R. Wheen. Solving double dummy bridge problems by exhaustive search. In D.N.L. Levy and D.F. Beal, editors, *Heuristic Programming in Artificial Intelligence – The First Computer Olympiad*, pages 89–94. Ellis Horwood, 1989.