SOLVING GO ON SMALL BOARDS

Erik C. D. van der Werf [1] *H. Jaap van den Herik* [1] *Jos W. H. M. Uiterwijk* [1]

Maastricht, The Netherlands

ABSTRACT

This article presents a search-based approach of solving Go on small boards. A dedicated heuristic evaluation function combined with the static recognition of unconditional territory is used in an alpha-beta framework with several domain-dependent and domain-independent search enhancements. We present two variants of the GHI problem (caused by super-ko rules) with some possible solutions. Our program, MIGOS, solves all small empty square boards up to 5×5 and can be applied to any enclosed problem of similar size.

1. INTRODUCTION

Many games have been solved using a search-based approach (van den Herik, Uiterwijk, and van Rijswijck, 2002). Go is a notable exception. In the last decade, Go received significant attention from AI research (Bouzy and Cazenave, 2001; Müller, 2002). Yet, despite all efforts, the best computer Go programs are still in their infancy. Although the game is played on a 19×19 board a considerable amount of research focuses on smaller problems, in particular on solving the game for small empty boards. Up to now, the largest square board for which a computer solution has been published is the 4×4 board (Sei and Kawashima, 2000). Some results based on human analysis exist for 5×5, 6×6 and 7×7 boards, but they are difficult to understand and so far they are not confirmed by computers (Davies, 1994, 1995b; van den Herik *et al.*, 2002).

This article presents a search-based approach of solving small Go problems. Our search method is the well-known alpha-beta framework extended with several domain-dependent and domain-independent search enhancements. A dedicated heuristic evaluation function is combined with the static recognition of unconditional territory to guide the search towards an early detection of final positions. Our program called MIGOS (MIni GO Solver) solves all square boards up to 5×5 and can be applied to any enclosed problem of similar size. To support the relevance of our research we quote Davies (1994).

> "*If you doubt that 5×5 Go is worthy of attention, you may be interested to know that Cho Chikun devoted over 200 diagrams to the subject in a five-month series of articles in the Japanese Go Weekly.*"

The article is organised as follows. Section 2 provides the rules of the game. Section 3 discusses the evaluation function. Section 4 deals with the search method and its enhancements. Section 5 presents an analysis of problems with super ko. Section 6 provides experimental results. Finally, Section 7 gives our conclusions.

2. RULES OF THE GAME

The game of Go is played by two players, Black and White, who consecutively place a stone of their colour on an empty intersection of a square grid. Usually the grid contains 19×19 intersections. However the rules

[1] Search and Games Group, IKAT, Department of Computer Science, Universiteit Maastricht, P.O. Box 616, 6200 MD Maastricht, The Netherlands. Email: {e.vanderwerf,herik,uiterwijk}@cs.unimaas.nl

are flexible enough to accommodate any other board size. Directly neighbouring (connected) stones of the same colour form a block. Stones remain fixed throughout the game unless their whole block is captured. The directly neighbouring empty intersections of (blocks of) stones are called liberties. A block is captured and removed from the board when the opponent places a stone on its last liberty. Moves that do not capture an opponent block and leave their own block without a liberty are suicide (if suicide is legal the own block is removed). Initially the board is empty, but as the game progresses some stable regions occur which are either controlled by Black or by White. A player is allowed to pass. The player who controls most territory in the end wins the game.

Although all major rule sets agree on the general idea stated above there exist several subtle differences (British Go Association, 2001). The main differences deal with the ko rule (repetition), life and death, suicide, and the scoring method at the end of the game which will all be discussed below.

2.1 The ko rule

Since stones can be captured (and removed from the board) it is possible to repeat previous board positions. However, infinite games are not practical, and therefore repetition of positions should be avoided. The most common case of a repeating position is the basic ko, shown in Figure 1, where Black captures the marked white stone by playing at position *a* after which White recaptures the black stone by playing a new stone at the marked position. The basic-ko rule says that direct recreation of a previous board position in a cycle of two moves is forbidden. As a consequence White can only recapture the black stone after playing a threatening move elsewhere which changes the whole-board position. Such a move is called a ko threat.

Under all rule sets the basic-ko rule applies. However, the basic-ko rule does not prevent repetitions of longer cycles. A common example of repetition in a longer cycle is the triple ko (three basic kos), but more complex positions with cycles of arbitrary length exist. Although such positions are rare they must be dealt with if they occur in a game. Rules for preventing long cycles are called super-ko rules. Unfortunately, there is no agreement among the various rule sets on the exact implementation of super-ko rules.

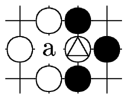

Figure 1: Basic ko.

In practise there are two questions that must be answered.

1. When is a position a repetition?

   First of all, for a position to be a repetition the arrangement of the stones must be identical to a previous position. However, there are more issues to consider than just the stones. We mention: the player to move, the points illegal due to the basic-ko rule, the number of consecutive passes, and the number of prisoners. When only the arrangement of stones on the board is used to detect repetition the super-ko rule is called *positional*, otherwise it is called *situational*.

2. What are the consequences of the repetition?

   The Japanese Go rules (Nihon Kiin and Kansai Kiin, 1989) state that when a repetition occurs, and if both players agree, the game ends without result. In the case of ‘no result’ humans normally replay the game. However if time does not permit this (for instance in a tournament) the result of the game can be treated as a draw (jigo). If players play the same cycle several times but do not agree to end the game, then as an extension to the rule they are considered to end it without result (Jasiek, 1997). For solving the game by computer ‘no result’ is not an option. Therefore, such repeated positions are scored drawn unless one side captured more stones in the cycle. In that case the player that captured the most wins the game. (If both sides would repeat the cycle sufficiently long one player could give away the whole board while still winning on prisoners.)

   An alternative used by several modern rule sets is to declare all moves illegal that recreate a previous whole-board position. The effect on the tree search is equivalent to saying that the first player to repeat a position directly loses the game with a score worse than the maximum loss of board points (any other move that does not create repetition is better).

   It should be noted that repetition created by a pass move is never declared illegal or drawn (passing at the end of the game must always be legal).

In this article we investigate the following three different compilations of the rules mentioned above.

1. **Basic ko** only prevents direct repetition in a cycle of two moves. Longer cycles are always allowed. If we do find a win (or loss) it means that all repetitions can be avoided by playing well.

2. **Japanese ko** is an extension of the basic-ko rule where repetitions that are not handled by the basic-ko rule are scored by the difference in number of pass moves in one cycle. For Japanese rules this is equivalent to scoring on the difference in prisoners captured in one cycle (since the configuration of stones on the board is identical after one cycle any non-pass move in the cycle must result in a prisoner). Repetition takes into account the arrangement of stones, the position of points illegal due to the basic-ko rule, the number of consecutive passes, and the player to move. This ko rule most closely reflects the Japanese rules, translating 'no result' to 'draw'.

   It should be noted that the Chinese rules (Davies, 1992) also allow repetitions to be declared drawn. However, that process involves a referee and is internally inconsistent with other rules stating that reappearance of the same board position is forbidden.

3. **Situational super ko (SSK)** declares any move that repeats a previous whole-board position as illegal. A whole-board position is defined by the arrangement of stones, the position of points illegal due to the basic-ko rule, the number of consecutive passes, and the player to move.

2.2 Life and death

In human games, the life and death of groups of stones is decided by agreement at the end of the game. Usually this is easy because a player only has to convince the opponent that the stones can make two eyes. If players do not agree they have to play out the position. For computers, agreement is not (yet) an option, so they always have to prove life and death. In practise it is done by playing until the end where all remaining stones that cannot be proved dead statically are considered alive. Static recognition of life and death has some consequences for the rules which will be discussed in Subsection 3.4.

2.3 Suicide

Though in most positions suicide is an obvious bad move, there exist positions where suicide could be used as a ko threat. In such cases it can be argued that suicide adds something to the game. A drawback of allowing suicide is that the length of the game can increase drastically if players are unwilling to pass (and admit defeat). In most rule sets (Japanese, Chinese, North American, etc.) suicide is not allowed (British Go Association, 2001). So, in all our experiments suicide is illegal.

2.4 The scoring method

When the game ends positions have to be scored. The two main scoring methods are territory scoring and area scoring. Territory scoring, used by the Japanese rules, counts the surrounded territory plus the number of captured opponent stones. Area scoring, used by the Chinese rules, counts the surrounded territory plus the alive stones on the board. The result of the two methods is usually the same up to one point. The result may differ because one player placed more stones than the other, either because Black made the first and the last move or because one side passed more often during the game. Another difference between the rule sets for scoring is due to the question whether points can be counted in so-called seki positions where stones are alive without having two eyes.

In all experiments reported in this article area scoring is used and points surrounded by a single player in seki count as territory. This type of scoring is usually referred to as Chinese scoring.

3. THE EVALUATION FUNCTION

The evaluation function is an essential ingredient for guiding the search towards strong play. So far there are neither good nor cheap evaluation functions for 19×19 Go (Bouzy and Cazenave, 2001; Müller, 2002). For small boards the situation is slightly better. We have been able to develop a decent evaluation function for small board Ponnuki-Go (a simplified version of Go) (van der Werf, Uiterwijk, and van den Herik, 2002). In this section we present a slightly modified version of the heuristic evaluation for Ponnuki-Go, and extend it with a method for early detection of sure bounds on the final score by recognising unconditional territory.

3.1 Heuristic evaluation

Our heuristic evaluation function aims at five goals: (1) maximising the number of stones on the board, (2) maximising the number of liberties, (3) avoiding moves on the edge, (4) connecting stones, and (5) making eyes. These goals relate in negated form to the opponent's stones. Since the evaluation function is used in tree search and is called in many leaves, speed is essential. Therefore our implementation uses bit-boards for fast computation of the board features.

Values for the first three goals are easily computed by directly counting relevant points on the board. It should be noted that instead of calculating individual liberties per block, the sum of liberties is directly calculated for the full board. The goals 4 and 5 (connections and eyes) are combined in one cheap estimation: the Euler number (Gray, 1971). The Euler number of a binary image is the number of objects minus the number of holes in those objects. Minimising the Euler number thus connects stones as well as creates eyes. Since the Euler number can be computed per two rows using a lookup table, only a small number of operations is needed. Eventually all values are linearly combined in a weighted sum to form the heuristic part of our evaluation function.

3.2 Static recognition of unconditional territory

For solving games a heuristic score is never sufficient. To be able to prove a win the highest and lowest values of the evaluation function must correspond to final positions (a sure win and a definite loss). For most games this is not a problem since the end is well defined (capture a piece, connect two sides etc.). In Go we face two problems.

The first problem is that most human games end by agreement, when both sides pass. For computers the end is usually detected by 2, 3, or 4 consecutive pass moves (the exact number of consecutive passes depends on the specific rule set). However, in nearly all positions where humans pass, a computer and especially a tree-search algorithm will try many more (useless) moves. Such moves do not affect the score and only increase the length of the game, thus pushing the final result over the horizon of any simple search.

The second problem is in the scoring itself. In even games the second player is usually given a number of so-called komi points, which are added to his score to compensate for the advantage of the initiative of the first player. In 19×19 Go the komi is usually between 5 and 8 points. For solving the game, without knowing the komi, it means that we have to determine the exact number of controlled points. The values for winning or losing are therefore limited by the maximum number of points on the board and should strictly dominate the heuristic scores. The score range we used for the 5×5 board is shown in Figure 2.

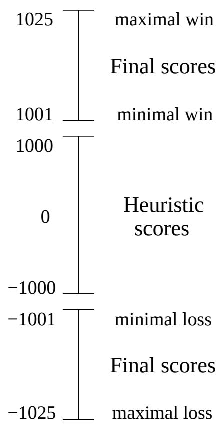

Figure 2: Score range for 5×5.

A requirement for provably correct results when solving the game is that the winning final scores are lower bounds (the worst that can happen is that you still win) and the losing final scores are upper bounds on the score, no matter how deep the tree is searched. For positions that are terminal (after a number of consecutive passes) this is easy, because there are no more moves. For other positions, which are closer to positions where humans would decide to pass, scoring is more difficult. To obtain reliable scores for such positions a provably correct analysis of life, death and unconditionally controlled territory is pivotal. Our analysis consists of 3 steps. First, we detect unconditional life. Second, we find the eyespace of possible unconditional territory. Third, we analyse the eyespace to see whether an invasion is possible.

Unconditional life

A set of stones is said to be unconditionally alive if they cannot be captured and never require any defensive move. A typical example of a set of unconditionally alive stones is a block with two small eyes. A straightforward approach to determine unconditional life would be to search out positions with the defending side always passing. Although such a search may have its merits it can easily become too costly.

A more elegant approach was developed by Benson (1976). His algorithm determines statically the complete set of unconditionally alive stones, in combination with a set of vital regions that form the eyespace. The algorithm is provably correct under the assumption that suicide is illegal (which is true for all major rule sets).

Benson's algorithm can be extended to determine safety under local alternating play (Müller, 1997). For strong (but imperfect) play the evaluation of life and death can be further extended using heuristics such as described by Chen and Chen (1999). Although some techniques discussed here are also used by Chen and Chen and by Müller, none of their methods are fully implemented in our program, first because heuristics do not suffice to solve the game, and second because relying on local alternating play is less safe than relying on an unconditional full-scope evaluation with global search.

Unconditional territory

Unconditional territory is defined as a set of points controlled by a set of unconditionally alive stones of the defender's colour where an invader can never build any configuration of uncapturable stones, even if no defensive move is made. The set of unconditionally alive stones, as recognised by Benson's algorithm, segments the board into the following three types of regions, illustrated in Figure 3, which form the basic elements for establishing (bounds on) the final scores.

1. **Benson-controlled regions** are formed by the unconditionally alive blocks and their vital regions (eyespace), as classified by Benson's algorithm. The surface of these regions is unconditional and directly added to the final score.

2. **Open regions** are not connected to unconditionally alive stones (which is common early in the game) or are between unconditionally alive stones of both sides. Since open regions can still be occupied by both sides they are played out further by the search.

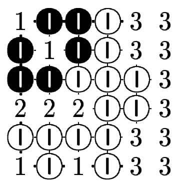

Figure 3: Regions to analyse.

3. **Closed regions** are surrounded by unconditionally alive stones of one colour. They may contain stones of any colour, but can never contain unconditionally alive stones of the invader's colour (because those would fall under type 1 and 2).

The rest of this subsection deals with classifying regions of type 3. For these regions we statically find the maximum number of sure liberties (usually eyes) an invader can make under the assumption that the defender always passes until the end of the game. If the maximum number of sure liberties is fewer than two the region is considered unconditional territory of the defending side that surrounds it (with unconditionally alive stones) and added to the final score. Otherwise it has to be played out. In the example of Figure 3 it means that, since the invader (Black) can build a group with two eyes in both corners (remember White always passes), the territory is not unconditionally controlled by White. A one-ply search will reveal that only one defensive move of White in region 3 (as long as it is not in the corner) is sufficient to make the full region unconditional territory of White.

To determine the maximum number of sure liberties each point in the interior of the region has to be classified as false or true eyespace. (False eyes are completely surrounded by stones of one colour but cannot provide sure liberties because they function as a connection.) Points that are occupied by the invader's stones are not considered as possible eyespace. Determining the status of points that form possible eyespace is done by counting the number of diagonally placed unconditionally alive defender stones. If the point is on the edge and no diagonally placed unconditionally alive defender stone is present then the point can become true eyespace. If the point is not on the edge (but more towards the centre) and at most one diagonally placed unconditionally alive defender stone is present, then it can also become true eyespace. In all other cases, except one, the eyespace will be false and cannot provide space for an eye. The only exception, in which a false eye is upgraded to a true eye, is when the false eye connects two regions that are already connected by an alternative path. This happens when the region forms a loop (around the unconditionally alive defender stones), which is easily detected by computing the region's Euler number.

An illustration of false eyes that are upgraded to true eyes is shown in Figure 4. All points marked *f* are initially false eyes of White (invader). However, all false eyes are upgraded to true eyes, since White might play both *a* and *b*, which is possible because we assume that Black (defender) always passes. (In practise Black will respond locally, unless there is a huge ko-fight elsewhere on the board.) If Black plays a stone on *a* or *b* the loop is broken and all points marked *f* become false eyes. The white stones and their neighbouring empty intersections then become unconditional territory for Black.

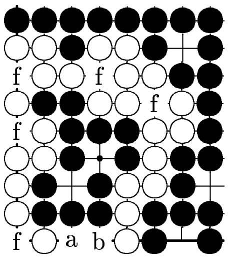

Figure 4: False eyes upgraded to true eyes.

Analysis of the eyespace

The analysis of the eyespace starts by looking for single defender stones. If a single defender stone is present on a false eye point then the directly neighbouring empty intersections cannot provide a sure liberty, and are therefore removed from the set of points that forms the true eyespace. (If a defender stone is connected to a second defender stone it may also remove an eye; however, this cannot be established statically and has to be determined by the search.)

Now that the true eyespace (henceforth called the eyespace) is found we test whether the eyespace is sufficiently large for two sure liberties. If the eyespace contains fewer than two points, or only two adjacent points, the territory is too small for a successful invasion and unconditionally belongs to the defender. If the eyespace is larger we continue with the analysis of defender stones inside the eyespace. Such stones may be placed to kill possible invader stones by reducing the size of the region to one single eye. In practise, we have to consider only two cases. The first case is when one defender stone is present in the invader's eyespace. The second case is when two defender stones are present in the invader's eyespace. If more than two defender stones are present in the invader's eyespace the territory can never be unconditional since the defender has to respond at least once to a capture (if he is able to prevent a successful invasion at all).

The analysis of a single defender stone is straightforward. The single defender stone contracts the stone's directly adjacent points of the invader's eyespace to a single eye, which provides one sure liberty. If the invader has no other region for eyes (non-adjacent points) any invasion fails and the territory unconditionally belongs to the defender.

The analysis of two defender stones in the invader's eyespace is harder. Here we start with noticing whether the two stones are adjacent. If the stones are non-adjacent they may provide two sure liberties; so, the territory is not unconditional. If the stones are adjacent they contract the surrounding eyespace to a two-point eye. If there is more eyespace, non-adjacent to the two defender stones, the area may provide two sure liberties for the invader and is not unconditional. If there is no more non-adjacent eyespace, the invader cannot be unconditionally alive (since he has at most one eye), but may still live in a seki together with the two defender stones.

Whether the two adjacent defender stones live in seki depends on the exact shape of the surrounding eyespace. If the two defender stones have three or fewer liberties in the eyespace of the invader, the region is too small for a seki and the area unconditionally belongs to the defender. If the two defender stones have four non-adjacent liberties and each stone directly neighbours two of the four liberties, the area can become seki. An example of such a position is shown in Figure 5. If the two defender stones have five liberties with more than one being non-adjacent, the area can become seki. If the two defender stones have six liberties the area can also become seki. Examples for five and six liberties can be obtained by removing one or two of the marked white stones in Figure 5. If White would play a or b the white group dies regardless of the marked stones. If one of the marked stones would move to c or d White also dies. If the area can become seki the two defender stones are counted as unconditional territory, but the rest is left undecided. If the area cannot become seki it unconditionally belongs to the defender.

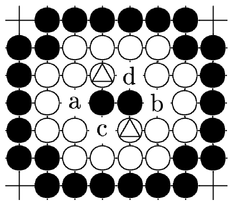

Figure 5: Seki with two defender stones.

3.3 Scoring terminal positions

In Go, games end when both sides stop placing stones on the board and play a number of consecutive passes. In all rule sets the number of consecutive passes to end the game varies between two and four. The reason why two consecutive passes do not always end the game is that the player first to pass might want to continue after the second pass. This typically occurs in positions where a basic ko is left on the board. If after two consecutive passes all moves are legal, the ko can be captured back. Therefore three or four consecutive passes are needed to end the game. The reason why under some rule sets four consecutive passes are required is that a pass can be worth a point, which is cancelled out by requiring an even number of passes. However, since passes at the end of the game do not affect area scoring, we require at most three consecutive passes.

In tree search the number of consecutive passes to end the game has to be chosen as restrictive as possible. The reason is that passes can otherwise push terminal positions over the search horizon. Therefore in the case that a position contains a basic ko, and the previous position did not contain a basic ko[2], the game ends after three consecutive passes. In all other cases two consecutive passes end the game.

Next, the terminal position has to be scored. Usually, many points on the board can be scored by recognising unconditional territory, as described in Subsection 3.2. However, not all territory is unconditional.

For scoring points that are not in unconditional territory dead stones must be removed from the board. This is done by counting the liberties of each block of stones. Each block that is not unconditionally alive and has only one liberty, which means it can be captured in one move, is removed from the board. All other stones are assumed alive. The reason why blocks with more than one liberty remain on the board is that they might live in seki, or could even become unconditionally alive after further play. If stones cannot live, or if the blocks with one liberty could have been saved, this will be revealed by (deeper) search.

Under situational super ko (SSK) the situation is more difficult. Blocks that are not unconditionally alive and have only one liberty are sometimes not capturable because the capture would create repetition (thus making the capture illegal). Therefore, under SSK, all non-unconditional regions must be played out and all remaining stones in these regions are assumed to be alive.

Once the dead stones are removed, each empty point is scored based on the distance toward the nearest remaining black or white stone(s)[3]. If the point is closer to a black stone it counts as one point for Black, if the point is closer to a white stone it counts as one point for White, otherwise (if the distance is equal) the point does not affect the score.The stones that remain on the board also count as points for their respective colour. Finally, the difference between black and white points, together with a possible komi, determines the outcome of the game.

3.4 Consequences for the rules

The approach used to determine unconditional territory, and (bounds on) final scores, contains a number of implicit assumptions about the rules. The first assumption is that suicide is illegal, which was already discussed in Subsection 2.3.

[2]This additional constraint prevents repetition after two consecutive passes in rare positions such as a double ko seki (Figure 7).

[3]In principle our distance-based area scoring can give slightly different results compared to other scoring methods when large open regions occur after removing dead stones of both sides based on having one liberty. However, the only position we found (so far) for which this might be considered a problem is a so called hane seki which does not fi t on the 5$\times$5 board.

The second assumption is that groups that have no space for two eyes or seki cannot live by an infinite source of ko threats elsewhere on the board. As a consequence moonshine life, shown in Figure 6, is statically classified dead if the surrounding stones are unconditionally alive. An example of an infinite source of ko threats is shown in Figure 7.

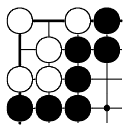
Figure 6: Moonshine life.

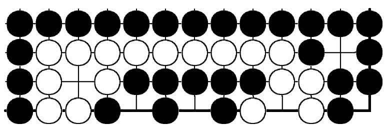
Figure 7: Infinite source of ko threats.

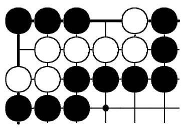
Figure 8: Bent four in the corner.

The third assumption is that groups that have possible space for two eyes or seki are not statically classified as dead. As a consequence bent four in the corner, shown in Figure 8, has to be played out. (Under Japanese rules the white group in Figure 8 is dead regardless of the rest of the board.)

We strongly suspect that the solutions for small boards (at least up to 5×5) are independent of the second and third assumption. The reason is that an infinite source of ko threats must be separated from another group by a set of unconditionally alive stones, which just does not fit on a small board. Nevertheless these assumptions must be noted if one would apply our system to larger boards.

The fourth assumption is that capturable stones surrounded by unconditionally alive blocks are dead and the region counts as territory for the side that can capture. As a consequence in a situation such as in Figure 9 Black controls the whole board, even though after an actual capture of the large block White would still be able to build a living group inside the new empty region. This conflicts with one of the inconsistencies of the Japanese rules (1989) by which the white stones are considered alive (though in practise White still loses the game because of the number of captured stones).

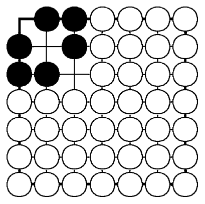
Figure 9: Capturable white block.

4. THE SEARCH METHOD

The standard framework for game-tree search is alpha-beta, which comes in many flavours. We selected the iterative-deepening Principal Variation Search (PVS) with a minimal window in a negamax framework (Marsland, 1986). The efficiency of the alpha-beta search usually improves several orders of magnitude by applying the right search enhancements. We selected the following enhancements: (1) transposition tables with enhanced transposition cutoffs, (2) symmetry lookups, (3) internal unconditional bounds, and (4) enhanced move ordering. All enhancements will be discussed below.

4.1 The transposition table

Transposition tables (Nelson, 1985) prevent searching the same position several times by storing best move, score, and depth of previously encountered positions. For the transposition tables we used the two-deep replacement scheme (Breuker, Uiterwijk, and van den Herik, 1996). Iterative-deepening search with transposition tables can cause some problems with super-ko rules to be discussed in Section 5. Enhanced transposition cutoffs (Plaat *et al.*, 1996) take additional advantage of the transposition table by looking at all successors of a node to find whether they contain transpositions that lead to a beta cutoff before a deeper search starts. Since enhanced transposition cutoffs are expensive they are only used three or more plies away from the leaves (where the amount of the tree that can be cut off is sufficiently large).

4.2 Symmetry lookups

The game of Go is played on a square board which contains eight symmetries. Furthermore, positions with Black to move are equal to positions where White is to move if all stones reverse colour. As a consequence these symmetries effectively reduce the state space by a factor approaching 16 (although in practise this number is significantly lower for small boards).

The effect of the use of the transposition table can be further enhanced by looking for symmetrical positions that have already been searched. In our application, when checking for symmetries, the hash keys for symmetrical positions are (re)calculated only when needed. Naturally this takes somewhat more computation time per symmetry lookup, but in many positions we do not need to look up all symmetries. In fact, since most of the symmetrical positions occur more near the starting position (where also the largest node reductions can be obtained) the hashes for symmetrical positions are only computed (and used) at a distance of 5 or more plies from the leaves. When multiple symmetrical positions are found, they are all used to narrow bounds on the score.

Since all symmetrical positions have to be reached from the root it is important to check the symmetries that are likely near the root. For empty boards all symmetries are reachable; however, for non-empty boards many symmetries can be broken. Time is saved by not checking unlikely symmetries in the tree.

It should further be noted that under SSK symmetrical transpositions can only be used for move ordering (because the history is different).

4.3 Internal unconditional bounds

Recognising unconditional territory is important for scoring leaves. However, in many cases unconditional territory can also be used in internal nodes to improve the efficiency of the search.

The analysis of unconditional territory, presented in Subsection 3.2, divides the board into regions that are either unconditionally controlled by one colour or are left undecided. The size of the regions can be used to compute upper and lower bounds on the score by assigning all undecided points either to Black or to White. If the maximum score is equal to or smaller than alpha, or the minimum score is equal to or larger than beta, the search directly generates a cutoff. In other cases the bounds can still be used to narrow the alpha-beta window, thus generating more cutoffs deeper in the tree.

Unconditional territory can further be used for reducing the branching factor. The reason is that moves inside unconditional territory normally do not have to be examined since they cannot change the outcome of the game. Exceptions are rare positions where changing the state just for the sake of changing the history for a future position is essential. Therefore all legal moves are examined under SSK.

4.4 Enhanced move ordering

The move ordering is enhanced by the following three heuristics: (1) history heuristic, (2) killer moves, and (3) sibling promotion. All move-ordering enhancements are implemented (and modified) to utilise the Go proverb “the move of my opponent is my move”.

The move ordering is as follows: first the transposition move, second the first move sorted by the history heuristic (Schaeffer, 1983), third the first killer move (Akl and Newborn, 1977), fourth the second move sorted by the history heuristic, fifth the second killer move, and finally the remainder of the moves sorted by the history heuristic. In positions that are searched sufficiently deep (in our experiments at least 5 plies away from the leaves) the move ordering is interleaved with sibling promotion (Dyer, 1995). After every move investigated, our implementation of sibling promotion takes the opponents expected reply as the next move to be investigated, unless the move is already examined or illegal. Killer moves rely on the assumption that a good move in one branch of the tree is often good in another branch at the same depth. The history heuristic uses a similar idea but is not restricted to the depth at which the moves are found. In our implementation the killer moves are stored (and tested) not only at their own depth but also one and two ply deeper. Further, our implementation of the history heuristic employs one table for both the black and white moves.

5. PROBLEMS WITH SUPER KO

Though the use of transpositions tables is pivotal for efficient iterative-deepening search it can create so-called Graph History Interaction (GHI) problems (Campbell, 1985) if the SSK rule applies. The reason is that the history of a position is normally not included in the transposition table, which means that in some special cases the transposition may suggest a continuation that is illegal or suboptimal under super ko. In our experiments we found two variations of the problem which we call the *shifting-depth variant* and the *fixed-depth variant*. Below both variants are presented, in combination with some possible solutions.

5.1 The shifting-depth variant

The shifting-depth variant of the GHI problem is illustrated by the following example. Consider an iterative-deepening search until depth n examining the following sequence of positions:

(root)-A-B-C-D-E-F-...-(heuristic score)

Now the heuristic score and the depth are stored in the transposition table for position D. Assume in the next iteration, the iterative-deepening process searches until depth $n+1$, and examines the following sequence of positions:

(root)-J-K-L-F-...-D

Here position D is found closer to the leaves. From the information stored in the transposition table D is assumed to be examined sufficiently deep (in the previous iteration). As a consequence the search directly returns the heuristic score from the transposition table. However, the result for D is not valid because the assumed continuation contains a repetition (F). This will not be observed since the moves are not actually made. The problem is even more disturbing because the transposition is found at another depth in a following iteration, which means that results of the next iterations can inherit a heuristic score, even though a final score would have been calculated if the moves were actually made.

It is possible to construct positions where alternating lines of play continuously inherit heuristic scores from previous iterations through the transposition table. An example of such a position, found for the 3×3 board, is shown in Figure 10. Here White's move 1 is a mistake under situational super ko because Black could take control of the full board by playing on *a* or *b*. However, the iterative-deepening search only returns heuristic scores because of depth-shifted transpositions (which assume continuations that are illegal under SSK). Black does not see the optimal win either, because of the same problem, and will play 2 obtaining only a narrow victory of one point.

To overcome shifts in depth we can incorporate the number of passes and captured stones in the full hash used to characterise positions. Then only transpositions are possible to positions found at the same depth.

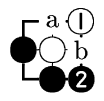

Figure 10: Suboptimal due to shifting depth.

5.2 The fixed-depth variant

Although fixing the depth (by including passes and captures in the hash) solves most problems, it still leaves some room for errors. The fixed-depth variant of the GHI problem is illustrated by the following example. Consider an iterative-deepening search examining the following sequence of positions:

(root)-A-B-...-C-...-B

Since B is illegal because of super ko this will become:

(root)-A-B-...-C-...-D-...

Now C is stored in the transposition table. Assume after some time the search investigates:

(root)-E-F-...-C

C is found in the transposition table and was previously examined at the same depth. As a consequence this line will not be expanded further. However, the value of C is based on the continuation C-...-D where C-...-B-... may give a higher score.

Another example of the fixed-depth GHI problem when using SSK is illustrated by Figure 11. Here the optimal sequence is as follows: (1-4) as shown, (5) Black captures at the marked point, (6) White passes, (7) Black at 1, (8) White passes (playing at 2 is illegal by SSK), (9) Black captures and wins by 17 points.

Now assume the following alternative sequence: (1) Black at 3, (2) White at 4, (3) Black at 1, (4) White at 2. The configuration of stones on the board is now identical to Figure 11. However, under SSK Black cannot play the same follow-up sequence and loses by 3 points.

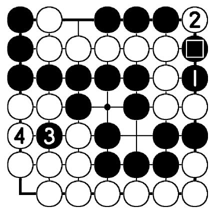

Figure 11: Black win by SSK.

Although both sequences of moves lead to the same position only the first sequence kills a white group. Since both sequences of moves lead to the same position at the same depth one of them, which one depends on the move ordering, can be valued incorrectly if the history is not taken into account.

In practise using separate hash keys for the number of passes and the number of captured stones, combined with a decent move ordering, is sufficient to overcome nearly all problems with super ko. However, for a proof this is not sufficient. Though it is possible to include the full history of a position in the hash, experiments indicate a drastic decrease in search efficiency, thus making it impractical for larger problems.

The reader should note that although (depth-shifted) transpositions are a problem for SSK they are not a problem under the Japanese-ko rule. The reason is that draws never dominate a win or a loss, because draws are in the range of the heuristic scores. In practise we use 0 as the value for draw, and we do not distinguish between draw and heuristic scores. (Alternatively one could use 1000 (or -1000) to indicate that Black (or White) could at least achieve a draw, which might be missed because of depth-shifted transpositions.) Occasionally depth-shifted transpositions are even useful, for solving positions under basic or Japanese ko, because they enable the search to look beyond the fixed depth of the iteration.

6. EXPERIMENTAL RESULTS

This section presents the results obtained by a Pentium IV 2.0 GHz computer, using a transposition table with 2^{24} double entries (for the two-deep replacement scheme). We discuss: (1) small board solutions, (2) opening moves on the 5×5 board, (3) the impact of recognising unconditional territory, (4) the power of search enhancements, and (5) preliminary results for the 6×6 board.

6.1 Small board solutions

MIGOS solved the empty square boards sized up to 5×5 [4]. Table 1 shows the ko rule, the best move, the result, the depth (in plies) where the PV becomes stable, the number of nodes, the time needed to find the solution, and the effective branching factor for each board. (In column 'time', s means seconds and h hours.)

The reader should note that 'depth' here does not mean the maximum length of the game (the losing side often can make some more futile moves). It just means that after that depth the Principal Variation and value of the move were no longer affected by deeper searches. As a consequence, boards that did not get a maximal score (e.g., 2×2 and 4×4), could in principle contain an undetected deep variant that might raise the score further. To rule out the possibility of a higher score both boards were re-searched with adjusted komi. The komi was adjusted so that it converted the loss of the second player to a win by one point.

[4] We would like to remind the reader that winning scores under basic ko are lower bounds for the score under SSK (because repetition can be avoided by playing well). Therefore the empty 5×5 board is solved regardless of super ko.

board	ko rule	Move	Result	Depth	Nodes (log_{10})	Time	b_{eff}
2×2	basic	a1	0	5	2.1	n.a.	2.65
	Japanese	a1	0	5	2.1	n.a.	2.65
	appr. SSK	a1	+1	11	2.9	n.a.	1.83
	full SSK	a1	+1	11	3.1	n.a.	1.91
3×3	basic	b2	+9	11	3.5	n.a.	2.06
	Japanese	b2	+9	11	3.5	n.a.	2.06
	appr. SSK	b2	+9	11	4.0	n.a.	2.30
	full SSK	b2	+9	11	4.4	n.a.	2.51
4×4	basic	b2	+1	21	5.8	3.3 (s)	1.90
	Japanese	b2	+2	21	5.8	3.3 (s)	1.90
	appr. SSK	b2	+2	23	6.9	14.8 (s)	1.99
	full SSK	b2	+2	23	9.5	1.1(h)	2.58
5×5	basic	c3	+25	23	9.2	2.7 (h)	2.51
	Japanese	c3	+25	23	9.2	2.7 (h)	2.51
	appr. SSK	c3	+25	23	10.0	9.7 (h)	2.73

Table 1: Solving small empty boards.

Finding the win then established both the lower and the upper bound on the score, thus confirming that they are indeed correct. Our results for boards up to 4×4 confirm the results published by van den Herik *et al.* (2002), which apparently assumed Chinese scoring with a super-ko rule. Since the two-point victory for Black on the 4×4 board is a seki (which is a draw under Japanese rules) it also confirms the results of Sei and Kawashima (2000). For all square boards up to 5×5, our results mutually confirm results based on human analysis (Tromp, 2002; Drange, 2002).

Table 1 shows results for two possible implementations of SSK. Approximate SSK does not check the full history, but does prevent most common problems by including the number of passes and the number of captured stones in the hash. Full SSK stores (and checks) a separate hash for the history in all entries of the transposition table. Both implementations of SSK require significantly more nodes than Japanese ko. In particular, full SSK (needed for a full proof) requires too much effort to be practical for larger boards.

It is interesting that under SSK Black can win the 2×2 board by one point. The search for this tiny board requires 11 plies, just as deep as solutions for the 3×3 board! Another conspicuous result is that under basic ko Black does not win the 4×4 board by two points. The reason is that the two-point victory is unreachable by a seki with a cycle where White throws in more stones than Black per cycle.

6.2 Opening moves on the 5×5 board

We analysed all opening moves for the 5×5 board. An opening in the centre leads to an easy win for Black, as shown in Figure 12a. However, alternative openings are much more challenging to solve. In Figure 13 the results are shown, with the numbered stones representing the winner (by the colour) and the score (by the number), for all possible opening moves.

Most difficult of all is the opening move on the (2,2) point. It is lost by one point only. After White's response in the centre MIGOS still required a 36-ply deep search (which took some days) to find the one-point victory. Proving that White cannot do better takes even longer. Optimal play for the (2,2) opening is shown in Figure 12c. Although this line of play is 'only' 21 ply deep, both Black and White can easily force much deeper variations. A typical example is when White throws in 16 at 19, followed by Black at 16, which leads to a draw. If Black plays 11 at 15 he loses the full board at ply 40.

The opening on (3,2) is also challenging, requiring a 28-ply deep search. Optimal play for the (3,2) opening is shown in Figure 12b. Extending with White 6 at 7 is a mistake and leads to a seki won with 5 points by Black (a mistake that was overlooked even by Cho Chikun (Davies, 1995a)).

The results for both alternative opening moves support the main lines of the human solutions by Ted Drange, Bill Taylor, John Tromp and Bill Spight (Tromp, 2002). (However, we did find some subtle differences deep in the tree, due to differences in the rules.)

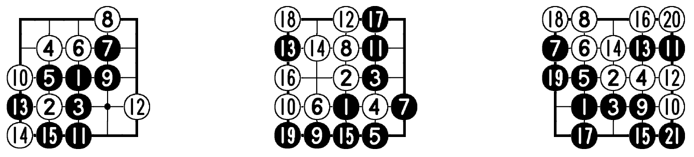
(a) (b) (c)

Figure 12: Optimal play for central openings.

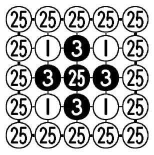
Figure 13: Values of opening moves on 5×5.

6.3 The impact of recognising unconditional territory

In Subsection 3.2 we introduced a method for static recognition of unconditional territory. It is used to detect final positions, or generate cutoffs, as soon as possible (thus avoiding deeper search until both players eventually must pass or create repetition). Although our method reduces the search-depth it does not necessarily mean that the search is always more efficient. The reason is that static analysis of unconditional territory is more expensive than a simple evaluation at the leaves (recognising dead stones by having only one liberty, or playing it out completely). To test the impact on the performance of recognising unconditional territory we used our program to solve the small empty boards without recognising unconditional territory, and compared it to the version that did recognise unconditional territory.

board	Use UT	Depth	Nodes (log_{10})	Time	Speed (knps)	b_{eff}
3×3	+	11	3.5	n.a.	n.a.	2.06
3×3	-	13	3.8	n.a.	n.a.	1.97
4×4	+	21	5.8	3.3 (s)	214	1.90
4×4	-	25	6.4	12.7 (s)	213	1.80
5×5	+	23	9.2	2.7 (h)	160	2.51
5×5	-	$>$30	$>$10.7	$>$2 (d)	$\sim$ 340	

Table 2: The impact of recognising unconditional territory.

The results, shown in Table 2, indicate that recognising unconditional territory reduces the search depth, the time, and the number of nodes for solving the small boards. (In column 'time', s means seconds, h hours and d days.) Although the speed in nodes per second is significantly less on the 5×5 board the reduced depth easily compensates this.

On the 4×4 board we observed no significant speed difference in nodes per second. For this there are at least four reasons: (1) most 4×4 positions cannot contain unconditionally alive blocks (therefore a large amount of costly analysis is often not performed), (2) many expensive evaluations are retrieved from a cache, (3) due to initialisation time the results for small searches may be inaccurate, and (4) the search trees are different (different positions require different time).

6.4 The power of search enhancements

The performance of the search enhancements was measured by comparing the decrease in number of nodes between a search using all enhancements and a search with one enhancement left out. The results, on the task of solving the various board sizes, are given in Table 3.

It is shown that on larger boards, with deeper searches, the enhancements become increasingly effective. The transposition table is clearly pivotal. Enhanced transposition cutoffs are quite effective, although the results suggest a slightly overestimated importance (because we neglect time). Symmetry lookups are very useful, at least on the empty boards. Internal unconditional bounds are not very effective because unconditional territory is also recognised at the leaves (on larger boards it may be better to turn off unconditional territory at the leaves and only use internal unconditional bounds). Our implementation of the history heuristic (one table for both sides) is very effective compared to the killer moves. This is also the reason why the first history move is examined before the killer moves. Finally, sibling promotion works quite well, especially on larger boards.

	3×3	4×4	5×5
Transposition tables	92 %	>99 %	>99 %
Enhanced transposition cutoffs	4 %	35 %	40 %
Symmetry lookups	63 %	89 %	86 %
Internal unconditional bounds	5 %	1 %	7 %
Killer moves	0 %	8 %	20 %
History heuristic	61 %	90 %	95 %
Sibling promotion	0 %	3 %	31 %

Table 3: Reduction of nodes by the search enhancements.

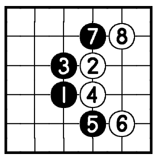

Figure 14: Black win (≥2).

6.5 Preliminary results for the 6×6 board

After solving the 5×5 board we tried to solve the 6×6 board. So far MIGOS did not solve the empty 6×6 board. However, based on human solutions it is possible to make the first few moves by hand. The most difficult position for which MIGOS proved a win, is shown in Figure 14. After about 13 days, searching 220 billion nodes in 25 ply, it proved that Black wins this position by at least two points. However, at this point we are not yet able to determine the exact value (which is expected to be a win by 4 points).

We tested MIGOS on a set of 24 problems for the 6×6 board published in 'Go World' by James Davies (1979; 1980). For 21 problems it found the correct move, for the other 3 it found a move which is equally good (at least for Chinese scoring). The correct moves usually turned up within a few seconds; solving the positions (i.e., returning a final score) took more time. MIGOS solved 19 problems of which two only reached at least draw (probably because of the occasional one-point difference between Japanese and Chinese scoring). All problems with more than 13 stones were solved in a few minutes or seconds only. The problems that were not solved in a couple of hours had 10 or less stones on the board.

7. CONCLUSIONS AND FUTURE EXPECTATIONS

The main result is that MIGOS solved Go on the 5×5 board for all possible opening moves. Further, the program solved several 6×6 positions with 8 and more stones on the board. The results were obtained by a combination of improved and new search enhancements together with a dedicated heuristic evaluation function and a method for static recognition of unconditional territory.

So far only the 4×4 board was solved (Sei and Kawashima, 2000). For this board their search required 14,000,000 nodes. MIGOS was able to confirm their solutions and solved the same board in fewer than 700,000 nodes. Hence we conclude that the static recognition of unconditional territory, the enhanced move ordering, the symmetry lookups, and our Go-specific improvements to the various search enhancements are key ingredients for solving Go on small boards.

We analysed the application of the situational-super-ko rule in tree search, and compared it to the Japanese rules for dealing with repetition. For solving positions, SSK quickly becomes impractical. It is possible to obtain good approximate results by reducing the information stored about the history of a position to the number of passes and captures. However, for most practical purposes super ko is irrelevant and can be ignored safely because winning scores under basic ko are lower bounds on the score under SSK.

Future expectations

The next challenges in small-board Go are: solving the 6×6 and 7×7 boards. Both boards are claimed to have been solved by humans, but so far no computer was able to confirm the results. The human solutions for the 6×6 board suggests a 4 point victory for Black (Tromp, 2002). The 7×7 board is claimed to have been solved by a group of Japanese amateurs including Kiga Yasuo, Nebashi Teruichi, Noro Natsuo and Yamashita Isao. In 1989, after several years of work, with some professional help from Kudo Norio and Nakayama Noriyuki, they reached the conclusion that Black wins by 9 points (Davies, 1995b).

On today's standard PC MIGOS is not yet ready to take on the empty 6×6 board. Although we might just sit back and wait a few years, there are ways to speed up the process, e.g., by using a massive parallel system, or by using the human solutions to guide and extend the search selectively, or work backward from the end.

On the AI side, we believe that large gains can be expected from adding more, provably correct Go knowledge to the evaluation function (for obtaining final scores earlier in the tree). Further, a scheme for selective search extensions, examining a highly asymmetric tree (resembling the human solutions), may enable the search to solve 6×6 and 7×7 much more efficiently than our current fixed-depth iterative deepening without extensions. Next to these suggestions an improved move ordering may increase the search efficiency, possibly even by several orders of magnitude.

Acknowledgements

We are grateful to Craig R. Hutchinson and John Tromp for information on the human solutions, and to Andries Brouwer, Ken Chen, Zhixing Chen, Robert Jasiek, the anonymous referees, and several others on the Computer-Go mailing list and rec.games.go for pointing our attention to some subtle details in the rules and providing feedback on earlier drafts of this article.

8. REFERENCES

Akl, S. G. and Newborn, M. M. (1977). The principal continuation and the killer heuristic. *1977 ACM Annual Conference Proceedings*, pp. 466–473, ACM, Seattle.

Benson, D. B. (1976). Life in the game of Go. *Information Sciences*, Vol. 10, pp. 17–29. ISSN 0020–0255. Reprinted in *Computer Games* (ed. D.N.L Levy), Vol. II, pp. 203-213, Springer Verlag, New York, 1988. ISBN 0-387-96609-9.

Bouzy, B. and Cazenave, T. (2001). Computer Go: An AI oriented survey. *Artifi cial Intelligence*, Vol. 132, No. 1, pp. 39–102. ISSN 0004–3702.

Breuker, D. M., Uiterwijk, J. W. H. M., and Herik, H. J. van den (1996). Replacement schemes and two-level tables. *ICCA Journal*, Vol. 19, No. 3, pp. 175–180.

British Go Association (2001), Comparison of some Go rules. http://www.britgo.org/rules/compare.html.

Campbell, M. (1985). The graph-history interaction: on ignoring position history. *Proceedings of the 1985 ACM Annual Conference on the Range of Computing: Mid-80's Perspective*, pp. 278–280, ACM, New York.

Chen, K. and Chen, Z. (1999). Static analysis of life and death in the game of Go. *Information Sciences*, Vol. 121, pp. 113–134. ISSN 0020–0255.

Davies, J. (1979). Small-board problems. *Go World*, Vol. 14-16, pp. 55–56.

Davies, J. (1980). Go in Lilliput. *Go World*, Vol. 17, pp. 55–56.

Davies, J. (1992). The rules of Go. *The Go Player's Almanac.* (ed. R. Bozulich), Ishi Press, San Francisco. http://www-2.cs.cmu.edu/∼wjh/go/rules/Chinese.html.

Davies, J. (1994). 5x5 Go. *American GO Journal*, Vol. 28, No. 2, pp. 9–12.

Davies, J. (1995a). 5x5 Go revisited. *American GO Journal*, Vol. 29, No. 3, p. 13.

Davies, J. (1995b). 7x7 Go. *American GO Journal*, Vol. 29, No. 3, p. 11.

Drange, T. (2002), Mini-GO. http://www.mathpuzzle.com/go.html.

Dyer, D. (1995). Searches, tree pruning and tree ordering in Go. *Proceedings of the Game Programming Workshop in Japan '95* (ed. H. Matsubara), pp. 207–216, Computer Shogi Association, Tokyo, Japan. http:/ /www.andromeda.com/people/ddyer/go/search.html.

Gray, S. B. (1971). Local properties of binary images in two dimensions. *IEEE Transactions on Computers*, Vol. C-20, No. 5, pp. 551–561. ISSN 0018–9340.

Herik, H. J. van den, Uiterwijk, J. W. H. M., and Rijswijck, J. van (2002). Games solved: Now and in the future. *Artifi cial Intelligence*, Vol. 134, Nos. 1–2, pp. 277–311. ISSN 0004–3702.

Jasiek, R. (1997), Commentary on the Nihon Kiin 1989 Rules. http://home.snafu.de/jasiek/j1989com.html.

Marsland, T. A. (1986). A review of game-tree pruning. *ICCA Journal*, Vol. 9, No. 1, pp. 3–19.

Müller, M. (1997). Playing it safe: Recognizing secure territories in computer Go by using static rules and search. *Proceedings of the Game Programming Workshop in Japan '97* (ed. H. Matsubara), pp. 80–86, Computer Shogi Association, Tokyo, Japan.

Müller, M. (2002). Computer Go. *Artifi cial Intelligence*, Vol. 134, Nos. 1–2, pp. 145–179. ISSN 0004–3702.

Nelson, H. L. (1985). Hash tables in Cray Blitz. *ICCA Journal*, Vol. 8, No. 1, pp. 3–13.

Nihon Kiin and Kansai Kiin (1989), The Japanese Rules of Go. http://www-2.cs.cmu.edu/∼wjh/go/rules/Japanese.html. Translated by J. Davies, diagrams by J. Cano, reformatted, adapted, and edited by F. Hansen.

Plaat, A., Schaeffer, J., Pijls, W., and Bruin, A. de (1996). Exploiting graph properties of game trees. *Proceedings of the Thirteenth National Conference on Articial Intelligence (AAAI'96)*, Vol. 1, pp. 234–239.

Schaeffer, J. (1983). The history heuristic. *ICCA Journal*, Vol. 6, No. 3, pp. 16–19.

Sei, S. and Kawashima, T. (2000). A solution of Go on 4x4 board by game tree search program, Fujitsu Social Science Laboratory. *The 4th Game Informatics Group Meeting in IPS Japan*, pp. 69–76 (in Japanese). Translation available at http://homepage1.nifty.com/Ike/katsunari/paper/4x4e.txt.

Tromp, J. (2002). Personal communication.

Werf, E. C. D. van der, Uiterwijk, J. W. H. M., and Herik, H. J. van den (2002). Solving Ponnuki-Go on Small Boards. *Proceedings of 14th Belgium-Netherlands Conference on Artifi cial Intelligence (BNAIC'02)* (eds. H. Blockeel and M. Denecker), pp. 347–354.