# *Alpha Beta, Pentalath*

***Tom Pepels***

*December 2012, Maastricht University*

Master of Artificial Intelligence, Intelligent Search and Games

***Abstract*** **– Pentalath is a computer-designed board game developed by Cameron Browne. This paper describes a Pentalath agent based on the alpha-beta framework. 6 Enhancements to the framework are proposed and tested. Moreover, an evaluation function with 6 features is discussed, and experiments run to determine its performance.**

## *1. Introduction*

This paper omits the details of alpha-beta search and the rules and definition of the game Pentalath. For a description of the game see: [4]. The paper is structured as follows: first, a description of the evaluation function is given and its 6 features explained. Next, the enhancements made to the alpha-beta framework are discussed. Lastly, experiments are detailed and discussed and a conclusion drawn. For the proposed program, an existing hexagon-library was used which can be found at [5].

## *2. Board evaluation*

The goal of Pentalath is to connect 5 stones on the hexagonal board in one row. To evaluate positions, the features under consideration should reflect this goal. The following features are used in the proposed evaluation function, followed by a detailed explanation of listed features:

1. The number of captured stones.
2. The longest 'free' row. Freedom denotes the possibility to extend the row to 5 stones.
3. The minimum freedom of all stones. In this case, freedom is calculated adjacent to groups of stones. As an indicator of the distance to a capture.
4. The size of the largest group of stones.

5. The maximum total directional freedom. This denotes in how many directions a player can expand his position to construct a row.
6. A side-to-play feature which subtracts a value based on whose turn it is.

**Captured stones:** When traversing the tree, each time a move is performed by the alpha-beta algorithm. A counter for each player is updated after every turn to hold the number of stones captured. This means that, at a leaf node the number of stones captured by each player is cumulated and weighed in the evaluation. After winning positions, captures are given the highest weight, as a capture in a game between strong players generally is a win for the capturing player.

**Longest free row:** Per player, each stone on the board belongs to at least one row. Since a row with a length of 5 constitutes a winning position, the distance to such a row is an important evaluator. However, only rows that can in fact be expanded without capture, to length 5 should be considered. Any other row, as long as it may be will have a high distance-to-win, unless it can be immediately captured. For white, the position depicted in figure 1 would result in a longest free row score of 2, since the row of size 3 was blocked by the black player.

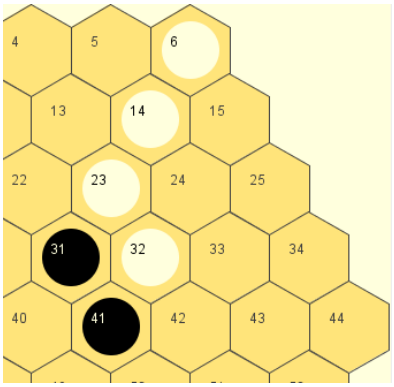


**Figure 1. White has longest row score: 2, Black similarly: 2**

*Odd/even effect:* The odd/even effect is strongly present in Pentalath. E.g. a row of length 3 or higher in the current turn is likely to be blocked by the opponent the next turn resulting in a considerably lower score. To prevent this inconsistent evaluation, we consider if the evaluated player is to move. If it is not the evaluated player's turn, the longest row for that player will always be assumed to be blocked on one side. This means that the freedom for that row to expand will be lower, and that the row will possibly not be considered in the evaluation.

*Mate detection:* Several moves in pentalath are guaranteed wins. Examples are shown in figures 2 to 4. Detecting these can predict a winning position several plies shallower than the actual winning position. In these cases, the longest row is given a score representing the remaining moves to win.

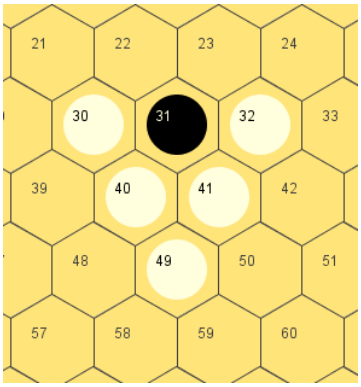


**Figure 2. White wins in 2**

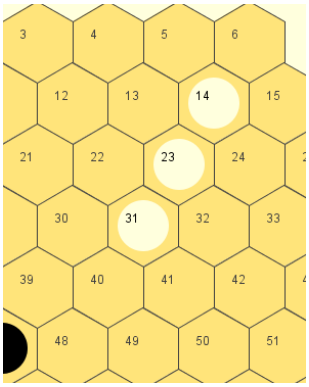


**Figure3. White wins in 2 if white to move**

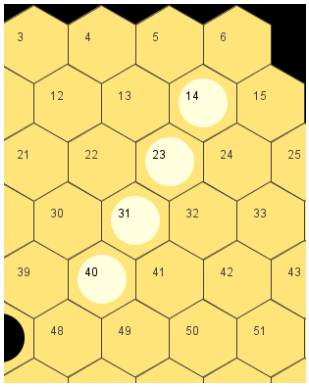


**Figure 4. White wins**

**Minimum freedom:** This feature represents a distance measure to the minimum number of moves required by the opponent to capture any of the player's stones. Pentalath features Go like capturing where surrounding the stones of your opponent completely removes them from the board. The freedom of a group of stones is defined as the number of free positions adjacent to the group. The minimum freedom then evaluates the worst-case scenario, and should be kept high to ensure the opponent will not capture any pieces. An example of the evaluation is depicted in figure 5. This feature is given a lower weight, as it is used to globally improve the strength of a player's position. It is not an indicator of the distance to win.

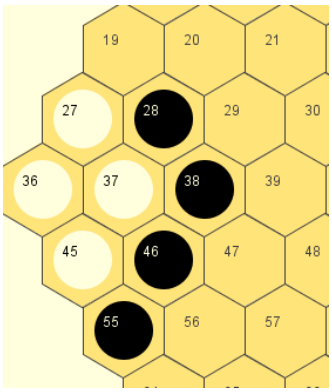


**Figure 1. White stones have freedom 1. Black has freedom 7**

**Largest group size:** A group is defined as any number of connected stones for a given player. Larger groups take more moves to capture. Moreover, constructing larger groups may lead to capturing of opponent pieces. Even better, connecting two groups may lead to a winning position.

This feature is used only after several moves have been made in the game, and is interchanged with the 'total freedom' feature. The feature is given a low weight as it is not a direct distance to win indicator. This feature also suffers from the odd/even effect, i.e. a group is likely to decrease in size as soon as the opponent makes a move. Therefore, if the player under consideration is to move, the size of the largest group is decreased by 1.

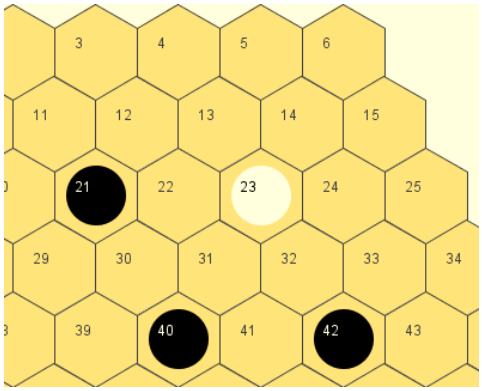


**Figure 2. White has total freedom 9**

**Maximum total freedom:** Positions at the start of a game require separate evaluation. Mostly, there are no opportunities for capturing, rows of both players are often of equal size, and groups aren't yet formed. An important aspect during the first moves is to have a position with ample opportunity to expand. Such a position is one with the most free positions in each direction. Figure 6 displays an example of this evaluation, here white can expand his position by 9. The feature does not take into account if the position can be expanded to a winning one. It merely accumulates the number of free positions in each direction and selects the highest. As this feature is only useful during the first moves, it is interchanged with the "largest group" feature after 6 moves have been made. It is given a high weight to ensure good opening positions and appropriate use of the swap-rule.

## *3. Alpha-Beta Enhancements*

Several enhancements for the alpha-beta framework are implemented in the proposed Pentalath agent. The main goal of these enhancements is to increase the search-depth at each move. This section discusses each of these in detail. Moreover, in the experimental results the impact on performance for each enhancement is detailed. Note that, the agent by default uses iterative deepening. As for tournament play, it is required to select a move within a strict time-limit.

The following enhancements will be discussed in this section:

1. Null moves
2. A transposition table
3. The relative history heuristic
4. Killer moves
5. Aspiration search
6. Static move-ordering

**Null move [1]:** This method attempts to reach more cut-offs by using a reduced search. Before executing normal alpha-beta search, a move is forfeited. The current player does not make any move and the turn shifts to the opponent. Next, search is performed at a reduced depth. If this reduced search produces a cut-off, we assume that the current position is strong and return the beta score. This is based on the heuristic that making a move will always improve one's position. In Pentalath this assumption is generally true, as increasing the number of stones on the board is never a liability. Though there may exist some specific positions in which this 'zugzwang' is present. A reduction of R = 2 was used in the proposed implementation. No consecutive null-moves are performed. Moreover, null-moves are performed starting at the 2nd ply.

**Transposition table:** The transposition table stores evaluations of previously evaluated positions. A hash is stored representing the gamestate, along with its value. In the proposed implementation, if a collision occurs, the position with the deepest subtree replaces the colliding entry [2]. A table of size 64 million positions is used. Each time a move is retrieved from the transposition table, the previously determined best move is played first when move-ordering.

**The relative history heuristic [3]:** The history heuristic is a general move-ordering method that can be used to improve alpha-beta performance in any game. For each ply in the search, moves that produce a cut-off are counted per player. Next, when ordering moves, after playing the killer move and transposition move, the remaining moves are ordered based on the counters. The relative history heuristic improves upon this strategy by ordering moves based on how often they were seen during the search, which is maintained on a so-called butterfly board. Moves are then ordered based on $hh\_score / bf\_score$. In the proposed implementation, both the bf_score and hh_score are incremented by 1, as proposed in [3].

**Killer moves:** Another move-ordering heuristic. Killer moves are based on the insight that moves that provide a cut-off on a certain ply are likely to produce a cut-off again. For this reason, cut-off moves

are stored per ply and used in move-ordering as the first and second move to play. That means that every time a cut-off is produced, the killer move for the current ply is updated. The move is stored as the first killer move; the previous killer move is then shifted to the second position.

**Aspiration search:** Generally, alpha-beta search is initially started with a window of $[-\infty, \infty]$. However, based on the previous score obtained by iterative deepening this window can be improved. To achieve this, after each search, alpha and beta are set to the value found - $\Delta$. In the proposed implementation $\Delta = 45$ was used. If the deeper search finds a new value within this bound, we can start the next iteration with the updated alpha and beta. However if the value lies outside the bounds, a new search is performed setting either alpha or beta to a default value of $-\infty$ or $\infty$, respectively.

**Static move ordering:** By default moves on the board are ordered stating in the center of the board and spiraling outward. Generally, moves played near the center of the board give the most freedom to expand and should therefore be considered first. Moves near edges appear more often in the end-game when less search-time is required, and can therefore safely be considered later.

## *4. Experimental setup*

Results are based on running 20 games for each experiment. The following values for constants were used:

- The depth decrease for null-moves: R = 2.
- $\Delta$ = 45 for aspiration search.
- Agents were allowed 15 seconds per move.
- A transposition table size of 64 Million positions.
- Butterfly board and history heuristic increment of 1.
- 2 Killer-moves per ply.

- A random value between 0 and 10 is added to the evaluation to ensure different moves are played between games.

**Weights:** By default, the features are weighed as follows for both players:

- Capture: **400** per captured stone
- Longest row: **40** per stone
- Minimum freedom: **5** per free, adjacent space
- Maximum total freedom: **5** per free square
- Largest group: **5** per stone
- Side-to-move: **-20 (if not current player's turn), 0 (otherwise)**

All experiments were performed on a quad-core AMD Phenom II X4 965 CPU with 8GB RAM.

## *5. Results*

The enhancements proposed in the previous sections were tested individually to determine their impact on performance. First, experiments using single enhancements enabled are discussed. These are paired against an (iterative deepening) alpha-beta implementation using the previously discussed evaluation function without enhancements. Next, each enhancement is individually disabled to determine their effect on performance against the proposed implantation using all enhancements and the discussed evaluation function. Each of the result-tables contains the fully enhanced and default alpha-beta as a reference.

| Enhancement | Win/loss | Average depth |
|---|---|---|
| Aspiration search | 12/8 | 5.4 |
| History heuristic | 11/9 | 6.1 |
| Killer moves | 14/6 | 6.1 |
| Null moves | 11/9 | 5.7 |
| Transposition table | 14/6 | 6.1 |
| No enhancements | - | 5.3 |
| All enhancements | - | 8.4 |

Table 1. Plain alpha-beta with single enhancements enabled

Table 1 displays the performance of individually enabled enhancements. Results show that the history heuristics in this case has the largest effect on the average depth reached. Aspiration search shows the least increase in this case, followed by the null-move heuristic. It is likely that null-moves have less gain since they benefit from move-ordering and the transposition table to ensure the extra search is performed as quickly as possible. This becomes apparent in Table 2, where the null-move heuristic performs on par with other enhancements. Moreover, aspiration search shows a promising win-loss rate when playing versus the non-aspiration program. This possibly implies that in some crucial cases a higher depth was reached due to the reduced window, but on average, the extra researches reach the same depth.

| **Enhancement *disabled*** | **Win/loss** | **Average depth** |
|---|---|---|
| Aspiration search | 7/13 | 8.0 |
| History heuristic | 7/13 | 7.5 |
| Killer moves | 6/14 | 7.6 |
| Null moves | 7/13 | 7.3 |
| Transposition table | 7/13 | 7.3 |
| No enhancements | - | 5.6 |
| All enhancements | - | 8.4 |

Table 2. Enhanced alpha-beta with single enhancements disabled

Considering the disabled cases in Table 2, we once again see the most increase in depth by the killer-moves enhancement. . In fact, based on these results, in the initial move ordering, killer-moves should be given a higher priority than the transposition table move.

| **Enhancement** | **Node reduction** |
|---|---|
| Aspiration search | 167,412 |
| History heuristic | 432,364 |
| Killer moves | 1,689,045 |
| Null moves | 1,097,151 |
| Transposition table | 476,032 |
| No enhancementsavg.: > 100.000.00 nodes | |
| All enhancements avg.: 3,965,386 nodes | |

Table 3. Node reduction per enhancement at ply 8

Table 3 shows the resulting node-reduction at ply 8. This result was obtained by, for each experiment, disabling a single enhancement and comparing its result with the fully-enhanced

program. The average number of nodes searched for the non-enhanced and fully enhanced algorithms are shown as reference. This result once more confirms the strength of the killer-move heuristic in the game, with the largest average node-reduction, followed by null-moves. Where the null-move heuristic showed the least increase in performance in the first experiment, in this case it is apparent that it benefits greatly from move-ordering and the transposition table, and is in fact able to reduce the search-space significantly.

Finally, the different features were tested and validated internally to determine their influence on performance. The results are shown in Table 4. For this test, each time, one of the non-crucial features was disabled and played 20 games against the default feature-set shown in the previous section. Only the program with the minimum freedom feature disabled seems to perform better than the default feature set.

| **Feature *disabled*** | **Win/loss** | **Average depth** |
|---|---|---|
| Total freedom | 8/12 | 8.0 |
| Max. group-size | 9/11 | 7.9 |
| Min. freedom | 11/9 | 7.8 |
| Max group-size, *always* use total freedom | 8/12 | 7.8 |

Table 4. Single features disabled

## *6. Conclusion*

Based on the experiments we can conclude that the proposed enhancements have a positive effect on the playing strength of alpha-beta. Moreover, when combined with other enhancements, the killer-move, and null-move heuristics have the largest impact on overall performance. Moreover, based on experimental results it was shown that the move-ordering priority for alpha-beta Pentalath players should be 1. Killer-Moves, 2. Transposition entry, and 3. History heuristic.

The program was entered in the Pentalath tournament hosted by Maastricht University on 21 December 2012. In which it defeated all but one of its opponents in a 7 round Swiss tournament. The resulting scores are listed in Table 5.

| Position | Name | Games | Score | Win | Draw | Loss |
|---|---|---|---|---|---|---|
| 1 | Tom Pepels | 7 | 13,0 | 6 | 1 | 0 |
| 2 | Oliver Trinnes | 7 | 12,0 | 6 | 0 | 1 |
| 3 | Benjamin Schnieders | 7 | 10,0 | 4 | 2 | 1 |

Table 5. Tournament results of the 7 round Swiss tournament [6]

## *References*


1. Goetsch, G. and Campbell, M.S., "Experiments with the null-move heuristic", *Computers, Chess, and Cognition*, 159 – 168, 1990.
2. Breuker, D.M. and Uiterwijk, J. and van den Herik, H.J., "Replacement schemes for transposition tables", *ICCA Journal*, vol. 17, no. 4, 183-193, 1994.
3. Winands, M. and van der Werf, E. and van den Herik, H. and Uiterwijk, J., "The relative history heuristic", *Computers and Games*, 262-272, 2006.
4. Browne, C. *Pentalath game description*, http://www.cameronius.com/games/pentalath/, accessed Dec. 2012.
5. Shestopalyuk, R, *Hexagon GUI*, https://github.com/silverio/samples, accessed Dec. 2012.
6. Uiterwijk, J, *Pentalath tournament results,* www.personeel.unimaas.nl/uiterwijk/Pentalath/, accessed Dec. 2012.