LEARNING THE PIECE VALUES FOR THREE CHESS VARIANTS

Sacha Droste and Johannes Fürnkranz[1]

Darmstadt, Germany

ABSTRACT

A set of experiments for learning the values of chess pieces is described for the popular chess variants Crazyhouse Chess, Suicide Chess, and Atomic Chess. We follow an established methodology that relies on reinforcement learning from self-games. We attempt to learn piece values and the piece-square tables for three chess variants. The piece values arrived at, are quite different from those of standard chess, and in several ways surprising, but they generally outperform the values that have been previously used in the literature, and in the implementations of computer players for these games. The results also underline the practical importance of piece-square tables for tactical variants of the game.

1. INTRODUCTION

Chess is arguably the most popular game in the western hemisphere, which is not only enjoyed in its standardized version but also in numerous variants (Pritchard, 2007). Chess variants can differ from standard chess in various ways. Here we mention six groups of variants. The variants may change (1) the board layout, (2) the number of players, (3) the movements of the pieces, (4) the pieces themselves, (5) the starting positions of the pieces, and (6) the dynamics of the game. Chess variants may have a long cultural tradition, like Japanese Chess (Shogi) or Chinese Chess (Xiangqi), or may have enjoyed a comparably recent popularity, such as Kriegspiel, where one only sees one's own pieces, composer Arnold Schönberg's Coalition Chess (Bündnis-Schach), a variant for four players played on a 10×10 board, or the 3-dimensional Star Trek Chess, which has enjoyed some popularity with the Star Trek TV series.

The rise of Internet Chess has also increased the popularity of certain chess variants. On chess servers like the *Free Internet Chess Server (FICS)*, various chess variants enjoy immense popularity. Group 6 of variants, most notably Fischerandom chess (Gligoric, 2003), differs from standard chess only through different starting positions. Conventional chess programs can be easily adapted to these variants (cf. Van Reem, 2006).

An seventh interesting group of chess variants is as follows. The standard set of pieces and their movements are maintained, but the dynamics of the game are changed in such a way that it can be expected that evaluation functions from standard chess cannot be re-used. For these variants, not even the value of the pieces is entirely clear. We provide five examples. (1) What is the value of a Queen in Suicide Chess, where the goal is to have captured one's own pieces as quickly as possible? (2) Does the high mobility make a piece good (because it can be sacrificed at various places) or bad (because it can be easily forced to capture)? (3) What is the value of a Pawn in hand in Crazyhouse Chess, where the captured pieces change colours and can at any time be re-inserted by the capturing side instead of making a move? (4) Sacrifices on squares such as f7 followed by a series of check-delivering pawn drops to force out the King are often the start of an irresistible attack in this chess variant. (5) What is the value of a Bishop in Atomic Chess, where captures cause explosions that remove every piece on a neighboring square?

In this article, we try to answer such questions by using machine-learning techniques to tune automatically evaluation functions for programs for these chess variants. In particular, we apply *temporal difference learning* to the problem of learning an evaluation function for three popular chess variants, viz. Crazyhouse Chess, Suicide

[1]Knowledge Engineering Group, TU Darmstadt, Hochschulstraße 10, D-64289 Darmstadt, Germany, email: juffi@ke.tu-darmstadt.de

Chess, and Atomic Chess. We closely follow the approach taken by Beal and Smith (1997, 1999a, 2001), who applied this methodology to Standard Chess and Shogi. However, while Shogi uses several pieces that move differently from regular chess, the variants that we consider in this article differ from Standard Chess only slightly. In particular, the basic movements of the pieces are the same for all chess variants considered, so that the results for these variants can easily be compared. One of our goals is to interpret intuitively the observed differences in the learned values, and see whether they can be explained by the differences in the rules of the chess variants.

The course of the article is as follows. We start with a description of the three chess variants (Section 2). Then we provide an overview of reinforcement learning in Standard Chess (Section 3). The experimental setup is given in Section 4. The Sections 5 to 8 each deal with a specific variant, viz. Standard Chess, Crazyhouse Chess, Suicide Chess, and Atomic Chess.

For each variant, we conduct two experiments. In the first experiment a simple material-only function is learned while the second experiment uses piece-square tables. The learned function will be analysed through (1) automated tournaments, and (2) intuitive explanations of striking aspects of the learned values. Section 9 contains conclusions.

Our work does not provide new methodological insights, but aims at determining piece values in these chess variants, since there are no well-established piece values. Thus, we believe that our answers, which are based on established statistical learning techniques, are of considerable interest to human and computer players of these games.

2. CHESS VARIANTS

We start with a brief introduction to the three chess variants that are discussed in this work, namely Crazyhouse Chess, Suicide Chess, and Atomic Chess. For each variant, we briefly sketch the rules of the game and discuss what aspects of the game are most interesting to us.

2.1 Crazyhouse Chess

Crazyhouse Chess is derived from the popular 4-player chess variant Bughouse Chess. In Bughouse Chess, two teams face each other on two boards, each team plays Black on one board and White on the other. The key twist is that if a player from Team A captures one of her opponent's pieces, she may pass the piece to her partner, who can, instead of moving one of her own pieces, place the captured piece on any free square on the board (at any time in the game). Simple communications between the partners (like "I need a Knight" or "Don't move!") are typically allowed. The game is won by the team that first delivers a mate on one of the two boards or lost by the team that first runs out of time on one of the two boards. Clocks are an essential part of this game, because deadlocks have to be avoided, where, e.g., each team has a mate-in-2 on one of the two boards and thus the respective opponents on this board refuse to move.

Crazyhouse Chess[2] is a two-player variant of this game. Whenever a player captures a piece, this piece changes its colour (from black to white or from white to black), and may be placed on the board at a later point in the game (alternative to a move). A common strategy is to collect multiple pieces in one's hand, and later to drive the opponents King out of his fortress with a number of successive piece placements. If a promoted piece is captured (i.e., a piece that once was a Pawn but was promoted to some other piece), it is transformed back to a Pawn. Because of the magical change of colours, this game cannot be played with standard chess pieces, but only virtually.[3] The game ends like regular chess, when one of the players mates the opponent or similar. Deadlocks cannot happen, and thus a clock is not necessary (but typically employed nevertheless).

Our main motivation for choosing this game is the unclear value of the pieces that a player holds in hand. For example, the combination of various Pawns and Knights in hand can often be a dangerous weapon because Pawns can often be placed next to a King and either drive the King out or yield a protected square next to the King which can be used for other Pawns or pieces, and Knights are the only pieces that can deliver check without being threatened by the King and without giving the opponent the chance to bar the check by dropping a piece.

[2] cf., e.g., http://wiki.wildchess.org/wiki/index.php/Crazyhouse

[3] It could be played with two sets of real chess pieces, but this may quickly become confusing. We are not aware of any real tournaments of this variant.

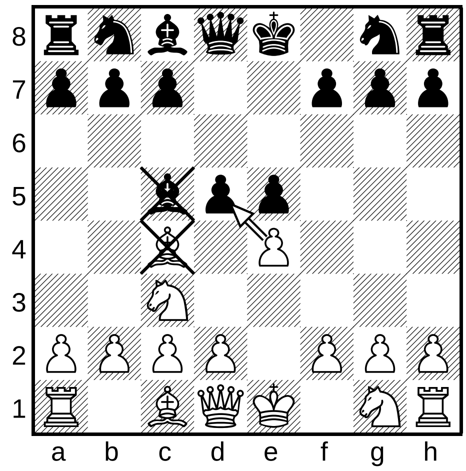


Figure 1: Explosions in Atomic Chess.

2.2 Suicide Chess

There are several variations of Suicide Chess (also known as Giveaway Chess on some chess servers).[4] They all have in common that the goal of the game is no longer to mate the opponent's King (and secondary goals are to take the opponents pieces) but to have captured one's own pieces. All pieces, including the King, move like in Standard Chess.

However, the King can be taken just like any other piece, checks do not have any significance and, in particular, do not restrict a player's move choices. Instead, any other possibility to take one of the opponent's pieces restricts one's move choices in the sense that one has to take a piece if this is at all possible (sometimes this has to be announced by the opponent, similar to a spoken "check"). If one can take several pieces, one can choose at will between those possibilities.

There are several possibilities for dealing with the situation when one player still has pieces, but no valid moves. According to the rules that are implemented at the *Free Internet Chess Server*, this is a win for the player that has fewer pieces on board. In case of a tie, the game ends in a draw.

Contrary to Standard Chess, it is unclear how valuable the pieces are. For example, one may believe that the Queen is a very bad piece because it offers the opponent many opportunities for placing a piece so that it may be captured. In contrast, it is also the piece with the largest number of move options, so that it can often allow the player to choose from multiple capture opportunities (which is better than having only one forced choice). It is also the piece that can be most easily disposed with. Here we remark that a Pawn does not provide many opportunities for an opponent, but is also quite hard to have it captured. There are no generally agreed-upon values for these and all other pieces of Suicide Chess. A general heuristic[5] says that Rooks are more valuable than Kings, Queens, and Bishops (they are all equal), which are in turn more valuable than Knights. Pawns are the least valuable pieces. As we will see, our learned values will slightly deviate from these specifications.

2.3 Atomic Chess

In Atomic Chess[6], capturing a piece leads to an explosion that extends from the square on which the piece is captured to all neighbouring squares (horizontally, vertically, or diagonally). The effect of the explosion is that, in addition to the captured piece, the capturing piece and all non-Pawns on neighbouring squares are removed from the board. For example, if White captures on d5 in Figure 1, the Pawns e4 and d5, as well as the Bishops on c5 and c4 will be taken off the board. The Pawn on e5 remains (explosions only concern pieces).

[4] cf., e.g., http://wiki.wildchess.org/wiki/index.php/Suicide
[5] http://wiki.wildchess.org/wiki/index.php/Suicide
[6] cf., e.g., http://wiki.wildchess.org/wiki/index.php/Atomic

Thus, there are direct and indirect threats to a piece: (1) a direct threat is a threat to capture the piece as in Standard Chess, (2) an indirect threat is the threat to capture one of its neighbouring pieces. Naturally, one can also lose by indirect mate, when a piece adjacent to the King is captured.

Again, there are minor variations of the game, which do not further concern us here. We used the variant that is played at the FICS.

Our main interest in this game is to see whether an increase in capturing power of each piece will have a significant effect on the piece values in comparison to Standard Chess.

3. REINFORCEMENT LEARNING IN STANDARD CHESS

For each of the considered chess variants (including Standard Chess), we automatically tuned an evaluation function via reinforcement learning. As the approach follows established techniques, we only briefly sketch the main ideas in this section, but refer to the literature for details. To provide adequate insight into the topic of reinforcement learning we deal with three topics. Features and evaluation functions are discussed in Subsection 3.1 The key idea of reinforcement learning is described in Subsection 3.2. An important enhancement is the idea on reinformed learning for deep searches (Subsection 3.3).

3.1 Features and Evaluation Functions

In our experiments, we use a straightforward linear evaluation function of the form

$$P(x) = \sum_f w_f \cdot f(x), \tag{1}$$

where $f(x)$ are elementary feature values that are computed for a given chess position x, and w_f are the weights. These weights are automatically tuned in the learning phase. For brevity, we omit the board x from the notation wherever it is clear from the context. We straightforwardly refer to a feature f. In this article, we consider two types of features: material and piece-square tables.

Material: For each piece type the feature type *Material* indicates the difference in the number of pieces of that type for White and for Black. In Standard Chess, there are five features (there is no feature for the King because both sides will always have exactly one King).

Piece-Square Tables: For each piece type and for each possible square on the board the feature type *Piece-Square Table* indicates, whether this square contains a piece of this type ($f = 1$) or not ($f = 0$). Different implementations of the piece-square table are possible, dependent on its use. For instance, we may have only a table for White, which is mirrored if Black is to move. In Standard Chess, there are $5 \times 64 + 48 = 368$ features of this type (please note (1) Kings included, and (2) piece-square tables for Pawns do not need the first and last row).

The basic features are combined with weights w_f, which reflect their relative importance. For example, in Standard Chess, a Knight typically counts three Pawns, i.e., the knight weight will be three times as high as the pawn weight ($w_N = 3 \cdot w_P$). Thus, if White has two Pawns for a Knight, the feature f_P will have the value 2 and the feature f_N will have the value -1, yielding a material position evaluation of

$$\begin{aligned} P &= w_P \cdot f_P + w_N \cdot f_N + w_B \cdot f_B + w_R \cdot f_R + w_Q \cdot f_Q = \\ &= 1 \cdot 2 + 3 \cdot (-1) + 3 \cdot 0 + 5 \cdot 0 + 9 \cdot 0 = -1 \end{aligned}$$

Determining the feature weights automatically from experience is the subject of this article. We note that the absolute value of the feature weights is irrelevant, only their relative values to each other determine which of a pair of positions is evaluated higher. However, the weights are typically normalized in terms of pawn values (i.e., $w_P = 1$).

For brevity, the above example only considered the material evaluation. If piece-square values are added, the number of features and corresponding weights increases, but the shape of the function remains the same.

3.2 Reinforcement Learning

The key idea of *reinforcement learning* (Sutton, 1988; Tesauro, 1992; Sutton and Barto, 1998) is to reinforce good moves by increasing the weights of features that contributed positively to the selection of this move, and by downgrading the weights of the features of the bad moves. Essentially, the *credit assignment problem*, i.e., the problem of deciding which moves in a game were bad and which ones were good, is straightforwardly solved by assuming that all moves in a won game were good, and all moves in a lost game were bad. If many million games are played, statistics will ensure that good moves will occur more frequently in won games than in lost games.

Temporal-difference learning is the reinforcement learning algorithm that is most frequently used in game playing. It is able to achieve a faster convergence by adapting the evaluation functions of a position at time t not towards the final outcome of the game, but towards the evaluation of the position at time $t + 1$. More precisely, the update rule of $TD(\lambda)$ is

$$\Delta W_t = \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} (\lambda^{t-k} \nabla_W P_k), \tag{2}$$

where α is a learning rate, P_{t+1} and P_t are two successive position evaluations, and λ is a parameter that influences how strongly positions P_k with $k < t$ should influence the weight adaptation of the current position. For example, $\lambda = 0$ means that only the last position P_t will have an influence, for $\lambda = 1$ all past positions of this game have the same influence.

For computing the weight update $\nabla_W P_k$, we first transform the linear evaluation function P' into a sigmoid function with values between 0 and 1. A typical function is

$$P(x) = \frac{1}{1 + e^{-\omega P'(x)}}, \tag{3}$$

where ω is a parameter that influences the steepness of the curve. For this type of function, the weight update for a single feature f of the linear function is

$$\nabla_W P_k = f \cdot P_k \cdot (1 - P_k). \tag{4}$$

3.3 Reinforcement Learning for Deep Searches

The greatest success story of reinforcement learning is Tesauro's (1992, 1995) TD-Gammon, which achieved world champion strength entirely by learning from self-play. However, Backgammon programs have a rather shallow search (three ply and no search extensions for TD-Gammon). In games like Standard Chess, deep searches are necessary for expert performance. A problem that has to be solved for these games is how to integrate learning into the search techniques. In particular in Standard Chess, one has the problem that the position at the root of the node often has characteristics that completely differ from the characteristics used at the evaluation of the node. The solution for this problem is to base the evaluation on the *dominant position* of the search. The dominant position is the leaf position in the search tree of which the evaluation has been propagated back to the root of the search tree (i.e., the position reached in the principal variation).

Consequently, one should adapt the weights of the dominant positions and not the weights of the root position to ensure that the estimation of the weight adjustments is based on the position that was responsible for the evaluation of the current board position. This problem has already been recognised and solved by Samuel (1959) but seemed to have been forgotten later on. For temporal-difference learning, this solution was rediscovered independently by Beal and Smith (1997) and Baxter et al. (1998b). Beal and Smith (1998) subsequently applied this technique for learning piece values in Shogi. A proper formalisation of the algorithm can be found in Baxter et al. (1998a, 2000, 2001).

For our experiments, we followed the description of the TDLeaf(λ) algorithm (Baxter et al., 1998c). The learning rate α was set using Temporal Coherence (Beal and Smith, 1999b). The key idea of this technique is to maintain a local learning rate α_f for each individual feature f, which degrades when the weight value for this feature stabilises (i.e., when the changes in the positive and the negative direction tend to equalise over time).

We will not further go into the technical details here, but refer to these original publications. A good overview of reinforcement learning in chess-like games can be found in Ekker (2003); surveys of learning approaches in computer chess are given in Skiena (1986) and Fürnkranz (1996); and for general overviews of machine learning in computer game-playing we refer to Fürnkranz (2001) and, more recently, Fürnkranz (2007) .

4. EXPERIMENTAL SETUP

As in most previous works, we used reinforcement learning to train an evaluation function from automated self-play only. It never played against a human player and no endgame or opening databases were used; i.e., no expert knowledge has influenced the experiments.

4.1 Learning Piece Values and Piece-Square-Table Values

For each of the considered chess variants (including Standard Chess), we conducted two experiments. Each experiment consisted of multiple runs, of which the results were averaged at the end.

The *small experiments* consisted of 10 runs of 2000 games each. Within these, a material-only evaluation function was learned, similar to the setup in Beal and Smith (1997) . The number of weights to be learned resulted from the chess variants. In Standard Chess and Atomic Chess, one value was learned for each piece except of the King, in Suicide Chess an additional value was learned for the King, and in Crazyhouse Chess additional values were learned for the pieces a player holds in the hand.

The *large experiments* consisted of 20 runs of 10,000 games each, and learned an evaluation function composed of material values and piece-square tables as described by Beal and Smith (1999a). In all of the examined variants a separate piece-square table was learned for every piece, including the King.[7] Beal and Smith (1999a) state that it depends on the phase of the game if the King's value is higher at the lower end of the board or in the centre. However, this does not necessarily have to be true for chess variants. For this reason (and to facilitate comparison later) the King received his own piece-square table in all of the variants. This lead to $6 \times 64 = 384$ additional weights in the large experiments.[8]

One could think of reducing the number of parameters by exploiting the vertical symmetry of the board. We deliberately did not do so, because in many of these chess variants the asymmetry between Kingside and Queenside is much more pronounced than in Standard Chess. For example, in Crazyhouse Chess, castling to the Queenside rarely ever happens, and much of the action focusses on attacks on the Kingside. Similarly, in Atomic Chess, attacks via f7 are deadly. As we will see, these dynamics are clearly visible from the learned piece-square tables.

4.2 Chess Engine

For our experiments, we adapted the open source chess engine SunSetter[9], which can deal with Standard Chess, Bughouse Chess, and Crazyhouse Chess. The program provides a straightforward implementation of standard computer-chess techniques (cf., e.g., Levy and Newborn, 1991) and uses a conventional alpha-beta search (Knuth and Moore, 1975) with quiescence search (Kaindl, 1982). It also has a primitive learning algorithm in the form of persistent transposition tables (Breuker et al., 1997). In all experiments, we used the chess engine with search depth 5, complemented with a quiescence search. Opening and endgame libraries were turned off.

We extended the chess engine, so that it is in addition able to play Suicide Chess and Atomic Chess, and, most importantly, integrates a reinforcement learning algorithm for learning feature weights as described in Section 3. The weights to be learned are all set to 1 at the beginning of each run, in accordance with Beal and Smith (1997) . To reduce (or even prevent) the repetition of games during the learning phase, which is necessary so that self-play

[7] In Crazyhouse Chess piece-square tables were learned only for pieces on the board, not for the placement of pieces in hand.

[8] 16 of these weights, those for Pawns on first and last rank, are irrelevant and will never change. For implementation reasons, we nevertheless kept them.

[9] http://sourceforge.net/project/showfiles.php?group_id=61676

does not converge towards repeating the same game over and over again, ties in the evaluation of possible moves in a given position are broken randomly. For the material-only evaluation, this happened quite frequently and was sufficient to avoid repeating games. In the experiments for learning piece-square tables, this was insufficient because the piece-square tables will give a different evaluation to almost every possible position on the board. Here, we slightly randomised the output of the evaluation function (about 1% of the evaluation value) in order to ensure a minimum level of exploration.

4.3 Evaluation

Our main interest is to see whether the learned values make sense to a chess player. To ensure objectively their quality, the learned values from both experiments for each variant are benchmarked in a concluding tournament. We have also directly compared the learned values to values found in the chess program DEEP SJENG, which is also able to play various chess variants.[10]. Following Beal and Smith, each pair of programs played 2000 games.

5. STANDARD CHESS

In order to ensure the correctness of our implementation, we also repeated the experiments in Standard Chess by Beal and Smith (1997, 1999a) . We report them here so that we can directly compare our results in the chess variants to those obtained for Standard Chess. For each game variant, we will start with the results of the two experiments (direct learning of piece values and learning via piece-square tables), and then try to interpret and compare the learned values.

5.1 Piece Values

Figure 2 shows the average course of the 5 values learned throughout the 2000 games played. Note how the values barely change after the 500-th game. This shows that the chosen length of 2000 games per run is sufficient The final average values with their standard deviations are depicted further below, in Figure 4.

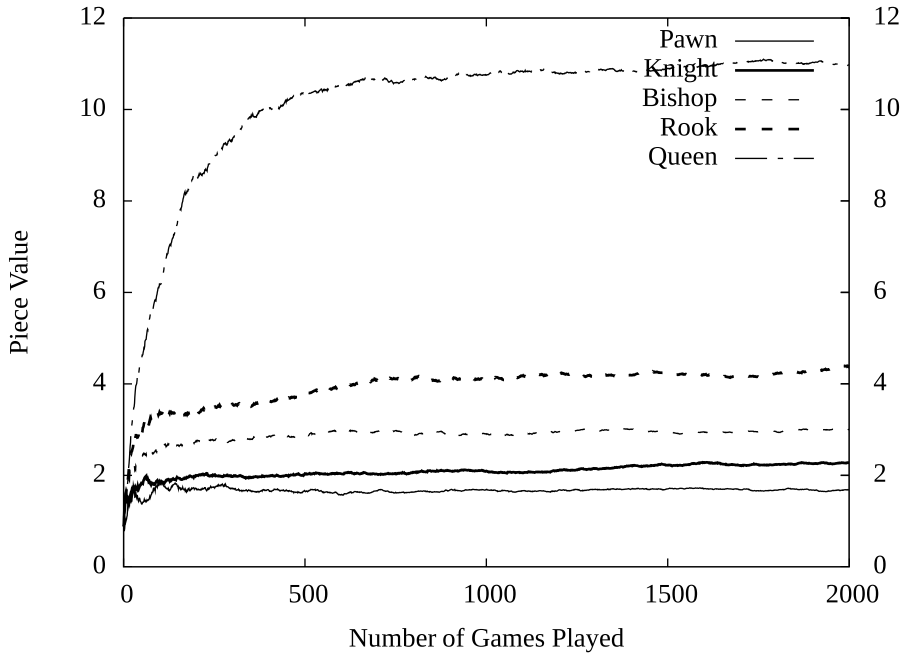


Figure 2: Development of material values.

[10]http://www.sjeng.org/indexold.html, we used version 11.2

For practical reasons, we have used a different normalisation for each chess variant. We will present our results in this normalisation, but summarise our results later on with the standard normalisation, where Pawns equal to 1 (Table 9). Here, we use a normalisation that scales the Bishop's value to 3.

5.2 Piece-Square-Table Values

The values of the learned piece-square tables are shown in Figure 3. The style of the six diagrams depicted follows the style suggested by Beal and Smith (1999a). In fact, the diagrams here look almost like an inversion of the diagrams in Beal and Smith (1999a) . The white square in a diagram designates the position at which the corresponding piece reaches its peak value. Accordingly, the black square shows the position with the lowest value. Underneath each diagram the minimum and the maximum values are given (i.e., the values of the white and the black square).

Figure 3.1 shows how the value of a Pawn increases with its rank on the board. The ranks 1 and 8 just hold their initial value since their squares are never occupied by a Pawn and are thus never modified. Figure 3.2 shows that a Knight's value decreases with increasing proximity to the boards edge. This can be explained by its reduced mobility. In contrast to the other officers (Bishop, Rook, Queen) the Knight has fewer valid moves on the border than in the middle of the board. Likewise the Bishop has lower values closer to border, however, looking at the minimum and maximum values, the effect does not have the same relevance here. The increased value of the Rook on the higher ranks is explained by Beal and Smith (1999a) by the ability to threaten the opponent's Pawns and King. The min and max values in the King's diagram are not the total values of the King but just the normalised entries in the piece-square table, since in Standard Chess the base value of the King is hypothetically ∞. The bright squares at the bottom of the diagram probably result from an advantageous position of the King in the middle of the game. The bright squares in the upper area (around f6) are probably due to an advantageous position during the endgame when it is important for the King to preserve his mobility and to avoid getting cornered.

5.3 Comparison of the Learned Values

The learned piece values and the piece-square tables were used to calculate average piece values. In this calculation the entries of the piece-square tables were weighted by the frequency of a piece visiting the according squares, again following Beal and Smith (1999a) . For this purpose the frequencies of the pieces visiting all the squares have been stored and summed up throughout the learning process. Figure 4 shows the average material values obtained from 20 runs. On top of each bar, we also show the standard deviation of the obtained estimate.

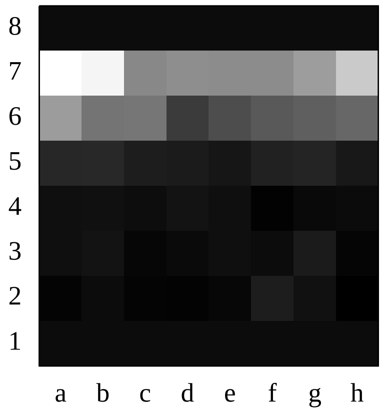
For comparison, the values from both experiments are shown in Table 1 together with those taken from Beal and Smith (1997, 1999; indicated by BS97 and BS99) and the original values from SUNSETTER C10. The values in Table 1, though, have all been normalised in the same manner to facilitate comparison.

	Avg. P.-Sq.	Piece Val.	SUNSETTER	BS97, Trial C	BS99, Full board
Pawn	1.05	1.20	0.96	0.96	0.97
Knight	2.82	2.43	2.87	2.14	2.69
Bishop	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
Rook	4.51	4.35	4.49	4.51	4.50
Queen	9.38	9.48	8.60	8.82	9.21

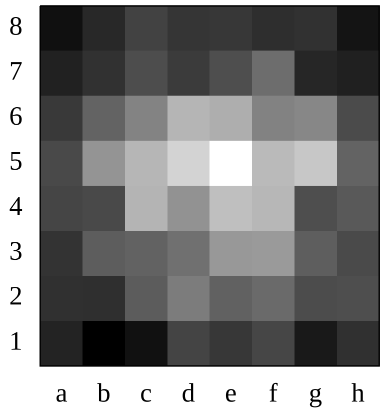
Table 1: Comparison of different sets of material values (Standard Chess).

The sets of values are all quite similar. Minor differences can be explained by the statistical nature of $TD(\lambda)$. Different games played during the learning process lead to slightly different values. This is, for example, confirmed by the different results of the trials A to E in Beal and Smith (1997) and the standard deviations shown in Figure 4. The results of the tournament comparing these values are given in Table 2. The entries in the configuration column have the following meanings.

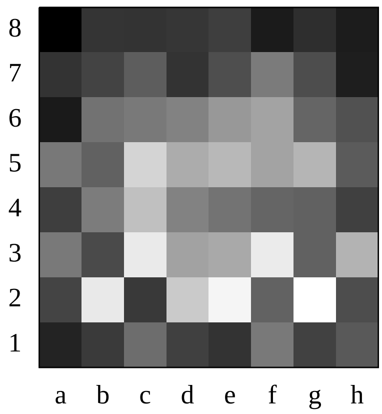
learned: material-only evaluation function using the learned piece values



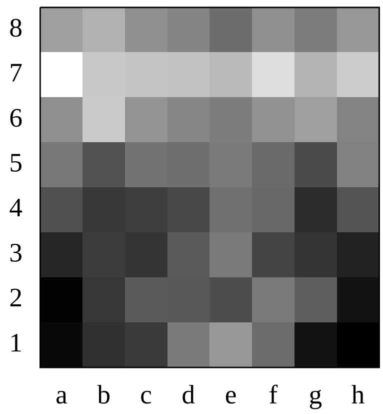
3.1: Pawn (Min/Max: 0.92/2.86)



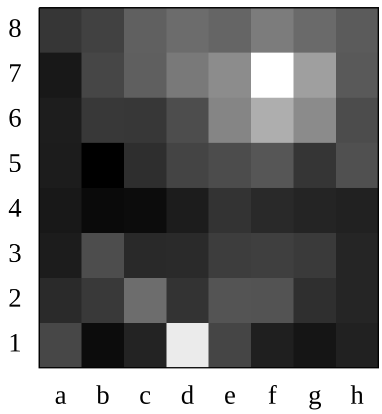
3.2: Knight (Min/Max: 2.61/3.13)



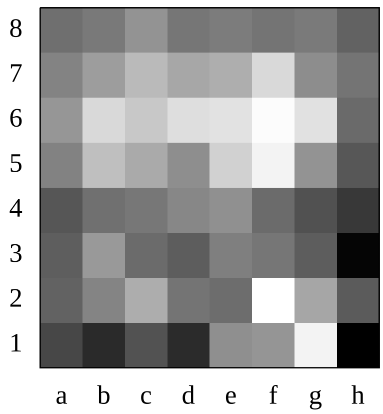
3.3: Bishop (Min/Max: 2.83/3.12)



3.4: Rook (Min/Max: 4.41/4.90)



3.5: Queen (Min/Max: 9.1/9.55)



3.6: King (Min/Max: 0.60/0.85)

Figure 3: Piece-square tables in Standard Chess.

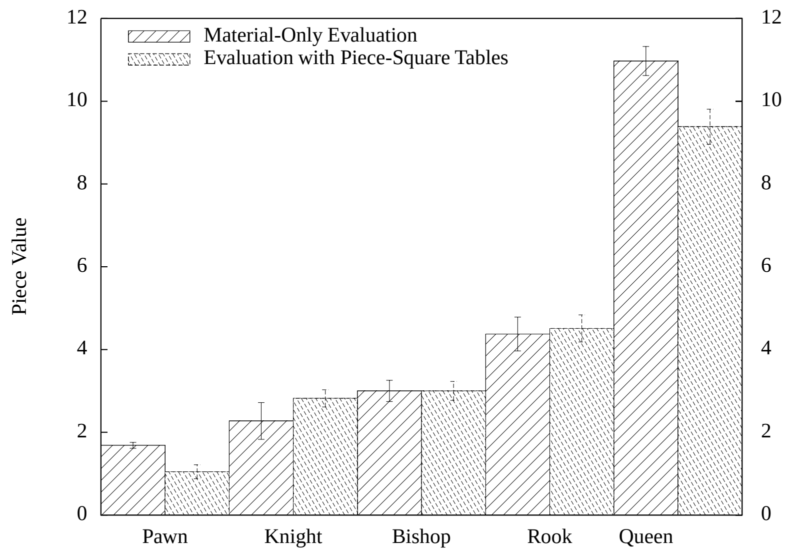


Figure 4: Final material values of the pieces (Standard Chess).

traditional: material-only evaluation function and traditional material values

original: material-only evaluation function and material values taken from SunSetter C10

average p-sq: material-only evaluation function and material values set to the average piece-square values of Table 1 (column Avg. P.-Sq.)

piece-square: the same evaluation function as above and the respective average learned configuration values

Deep Sjeng: the chess engine Deep Sjeng[11] version 11.2

Configuration	Won	Lost	Draw	Ratio
learned vs. traditional	1246	715	39	63.28%
learned vs. original	1037	906	57	53.28%
learned vs. average p-sq	971	993	36	49.45%
piece-square vs. learned	1735	245	20	87.25%
piece-square vs. Deep Sjeng	2	1997	1	0.13%

Table 2: Results of the tournament for Standard Chess.

The results show how the learned values actually lead to stronger play than values retrieved by other means. They also show that the engine plays significantly stronger with piece-square tables than with a material-only evaluation function. It is interesting to see that the winning ratio of learnt values against the averaged piece-square table values is slightly less than 50%. The last line shows a comparison to a fully fledged chess engine, and clearly demonstrates that (not surprisingly) piece-square tables alone are sufficient for a strong chess evaluation function.

6. CRAZYHOUSE CHESS

Section 6 contains three subsections, viz. piece values (6.1), piece-square-table values (6.2) and a comparison of the learned values (6.3).

[11] http://www.sjeng.org/indexold.html

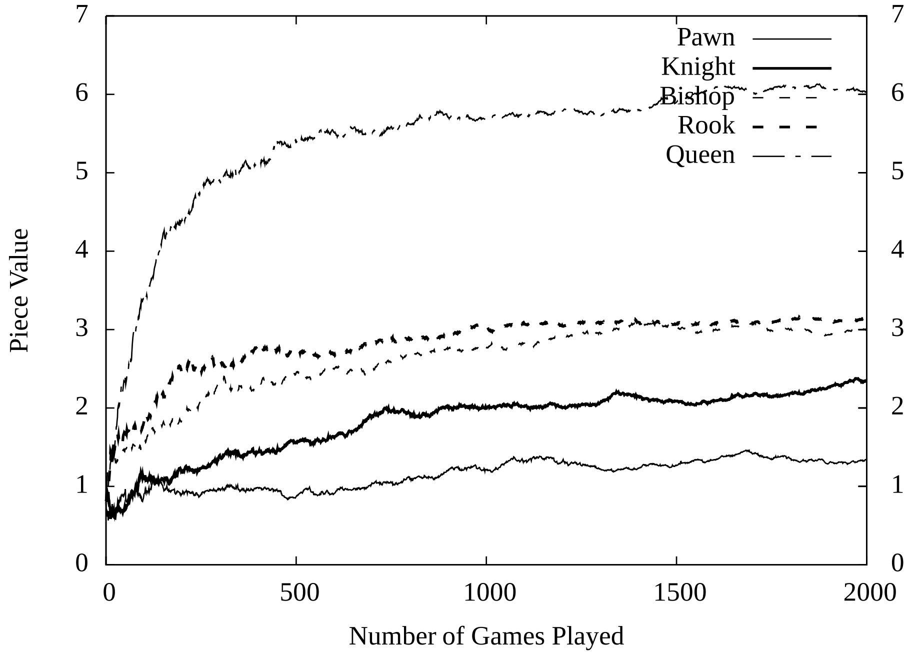


Figure 5: Development of piece values on the board (Crazyhouse).

6.1 Piece Values

The development of the average learned values for Crazyhouse Chess is shown in Figure 5. The values have been normalised analogously to the values in Standard Chess, i.e., a Bishop on the board has a value of 3.

The diagrams of Figure 7 resemble those from Standard Chess. The general appearance of the curves in Figure 5 is in accordance with those of Figure 2 and the values of Pawns and Bishops have barely changed. However, both of the stronger pieces Rook and Queen have significantly decreased in value. The Queen's value has almost halved, while the Rook has almost the same value as the Bishop now. The low value of the Queen probably results from the generally higher mobility on the board which is caused by the dropping moves. In comparison with these teleportation moves, the Queen's natural high mobility does not look so impressive anymore.

The development of the on-hand pieces' values is shown in Figure 6. They have also been normalised with the on-board bishop value, i.e., their final values are directly comparable to the on-board values. In particular Knights, Rooks, and Pawns are worth more when in the hand than when they are on the board. This is consistent with our practical experience with this game. In all likelihood the effect is caused by the strong correlation between a piece's position on the board and its value. The learned piece-square-table values will show whether this theory holds, since in that case the difference between the highest and the lowest value of a specific piece should be rather large.

It should also be noted that we made the simplifying assumption that the value of an in-hand piece is independent of the other pieces that the player is holding. This is almost certainly not the case. If we do not hold any Pawns yet, the value of a new Pawn in hand will typically be higher than if we are already holding three Pawns.

Interestingly, the convergence of the in-hand values is slower than in Figure 5. The reason for the slow convergence is the fact that these features seldomly have a value different from 0. The reason is, for example, that the Queen usually will be played on the board again immediately after it has been taken to the hand.

6.2 Piece-Square-Table Values

Figure 7 shows the learned piece-square tables for Crazyhouse Chess. The diagrams resemble those from Standard Chess, but there are still some apparent differences of which the explanation is based again on in-hand

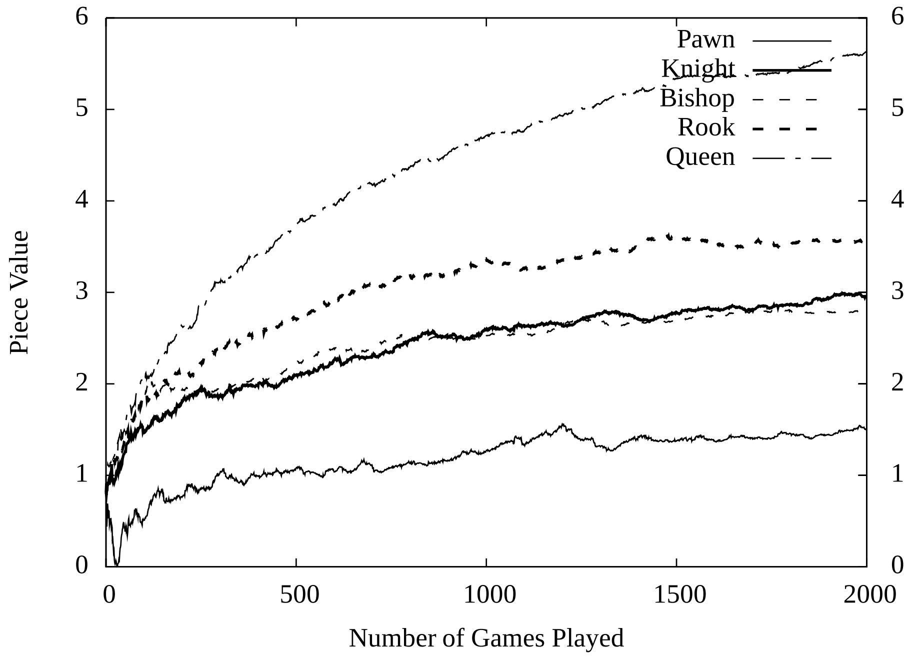


Figure 6: Development of material values of on-hand pieces (Crazyhouse).

pieces and dropping moves. First, we can see that the asymmetry between the Kingside and Queenside is much more pronounced than in Standard Chess, and that for all pieces, there is a clear preference for the Kingside. This reflects the fast dynamics of the game, where castling to the Queenside rarely ever happens (even castling is considerably rarer than in Standard Chess). Castling would typically have to be followed by a King's move to b1 to eliminate the imminent danger of a Rook or Queen to be dropped on a1, and thus such castling loses much more time than castling to the Kingside. As a consequence, pawn drops, a major attacking device in this game, almost never happen on the Queenside, but always near the opponent's King (to lure him out) or near the own King (to increase his protection and prevents piece drops on these places). Essentially, similar patterns can be observed for all pieces.

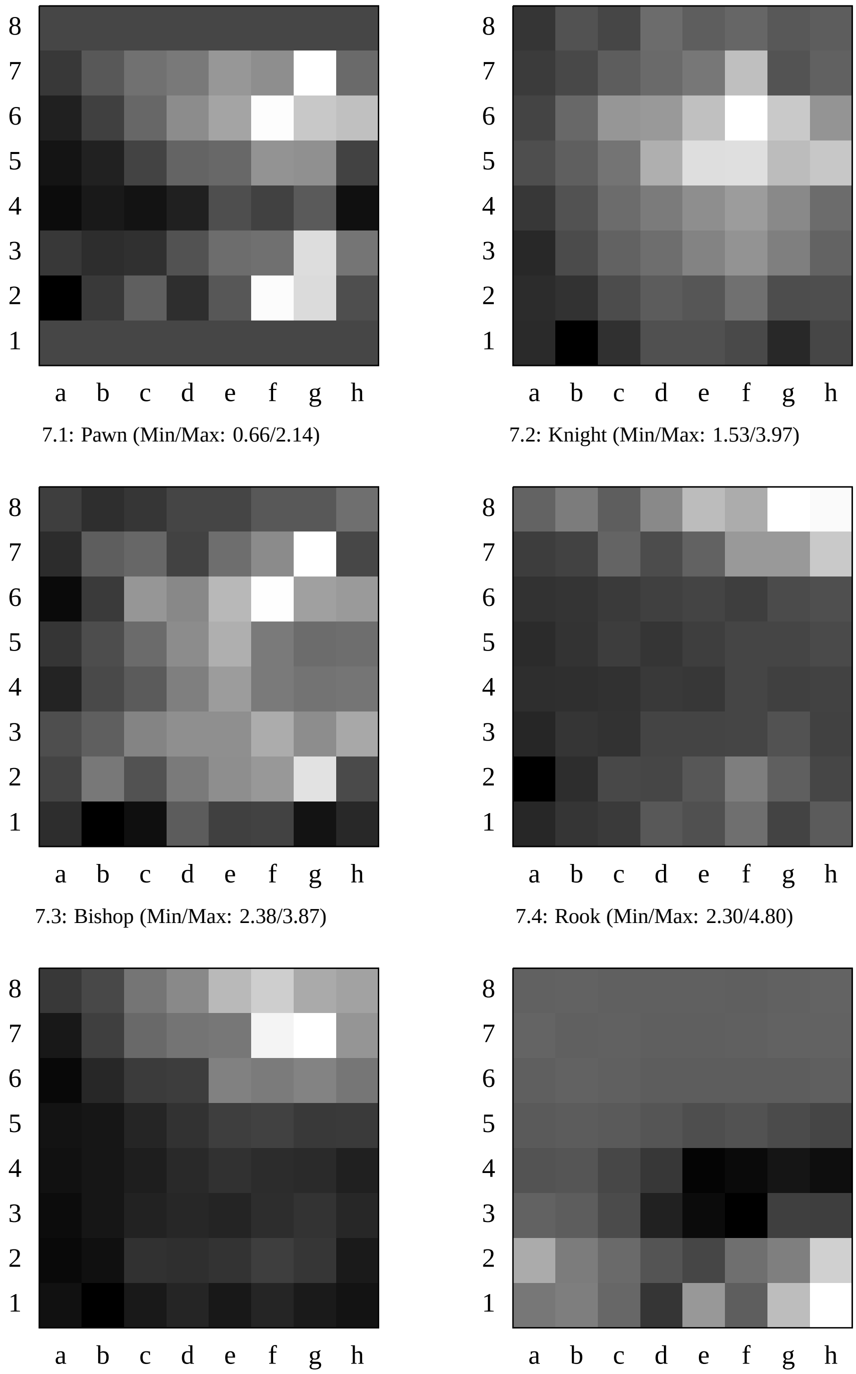
The pawns diagram differs particularly strongly from its Standard Chess counterpart. Since a Pawn can be placed on the seventh rank via a dropping move its value does not necessarily increase with its rank. The high values on the higher ranks can still be explained by the imminent promotion. However, the high values on the lower ranks show the increased importance of king safety. In particular the f-Pawn is clearly discouraged from moving, because doing so will give the opponent the chance to start a dangerous attack with a pawn drop on f2.

The most valuable square for the Knight (Figure 7.2) has moved from the centre position e5 to f6, where it immediately threatens the starting position of the opponent's King. A Knight's check from f6 is often the start of a deadly attack, and clearly this is also the favourite spot for a dropping move.

The recognition of any apparent pattern in Figure 7.3 is difficult. It is notable, though, that the difference between the lowest and the highest value is larger than in Standard Chess. This leads to the assumption that the absolute position of the Bishop on the board is more important in Crazyhouse Chess than it is in Standard Chess. The strong preference for fianchettos (i.e., Bishops placed on the squares b2 and g2) has somewhat diminished, which is also consistent with our personal experience (although we would think that the weight for a Bishop on g2 is still too high considering its vulnerability to pawn droppings on f3 or h3. One can also see that bishop droppings on g7 and f6 (attacking the Rook and penetrating the enemy's Kingside) are quite good.

The Rook's domain of usefulness on the board is now restricted from the upper ranks in general to the upper right corner of the board (g8 and h8), where it can be expected to do the most harm to the opponents king safety.

The Queen, too, has undergone a comparable change. Its diagram still has a bright area in the upper right part of the game board. However, the square d1 has lost its high value and vanishes into a generally dark zone. Just like the Bishop the Queen's difference in value between the best and the worst square has strongly increased.



7.1: Pawn (Min/Max: 0.66/2.14)

7.2: Knight (Min/Max: 1.53/3.97)

7.3: Bishop (Min/Max: 2.38/3.87)

7.4: Rook (Min/Max: 2.30/4.80)

7.5: Queen (Min/Max: 4.38/6.91)

7.6: King (Min/Max: -0.97/4.25)

Figure 7: Piece-square tables in Crazyhouse Chess.

The King is clearly safest in the corner, where it has the least number of neighbouring squares that need to be protected from drop pieces. The large grey area in the top half of Figure 7.6 results from the abolition of the endgame. Since no piece can ever permanently leave the board, no position arises that corresponds to the usual endgame in Standard Chess. The bright area from Figure 3.6 does not reappear here and the upper ranks (5 to 8) are barely used by the King. For this reason these squares' values remained practically unchanged.

6.3 Comparison of the Learned Values

The average material values for pieces on the board that have been calculated from the piece-square tables are shown in Figure 8, together with their standard deviation.[12] The same normalisation has been used as in Section 5 for Standard Chess.

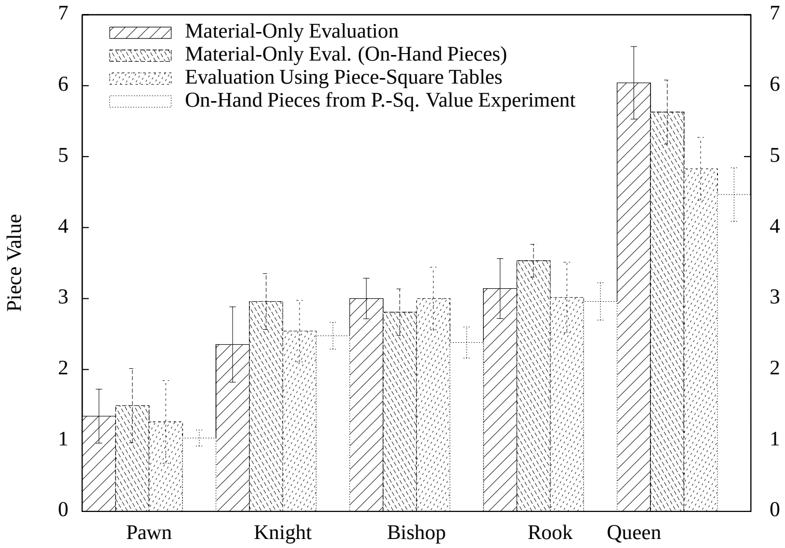


Figure 8: Final material values of the pieces (Crazyhouse Chess).

Table 3 compares the learnt piece values, the averaged piece-square-table values, the values from SUNSETTER 7e, and the values used by DEEP SJENG.

	Avg. P.-Sq.	Piece Value	SUNSETTER	DEEP SJENG
Pawn	1.26	1.34	1.54	1.30
Knight	2.54	2.35	2.95	2.74
Bishop	**3.00**	**3.00**	**3.00**	**3.00**
Rook	3.02	3.14	3.08	3.26
Queen	4.83	6.04	6.00	5.87
Pawn (Hand)	1.03	1.49	1.54	1.30
Knight (Hand)	2.48	2.96	2.95	2.74
Bishop (Hand)	2.38	2.81	3.00	3.00
Root (Hand)	2.96	3.53	3.08	3.26
Queen (Hand)	4.47	5.63	6.00	5.87

Table 3: Comparison between different sets of material values (Crazyhouse Chess).

The results of the tournament are presented in Table 4. The entries in the configuration column have the following meanings.

learned : material-only evaluation function using the learned piece values

[12]The values of the on-hand pieces correspond (normalised) to the ones learnt. No calculational blending with piece-square tables of any kind has been performed.

traditional : material-only evaluation function and traditional material values; on-hand pieces have the same values as pieces on the board.

original : material-only evaluation function and material values taken from SUNSETTER 7e

DEEP SJENG (Values) : material-only evaluation function and material values taken from file eval.c from DEEP SJENG

average p-sq : material-only function and material values set to the calculated ones in Table 3

piece-square : the same evaluation function as above and the respective learned configuration values

Configuration	Won	Lost	Draw	Ratio
learned vs. traditional	1352	646	2	67.65%
learned vs. original	1317	680	3	65.93%
learned vs. DEEP SJENG (Values)	1322	674	4	66.20%
learned vs. average p-sq	813	1184	3	40.73%
piece-square vs. learned	1928	71	1	96.43%

Table 4: Results of the tournaments (Crazyhouse Chess).

With a win ratio of about 66% the first three lines show that even a small optimisation of piece values can lead to stronger play. The observation that we made for Standard Chess, that the learned values perform worse than those that are calculated as weighted averages of the piece-square tables, aggravates to a degree which cannot solely be explained by statistical effects.

The win ratio of the program using piece-square tables even surpasses the ratio from the according Standard Chess tournament. This is in compliance with the increased differences between the minimum and maximum values in Figure 7. It confirms that due to the high mobility (pieces in hand can be placed at any square on the board), good piece placement gains importance, and thus piece-square tables are even more important for Crazyhouse Chess than they are for Standard Chess. This is also witnessed by the increase of the span between the minimum and maximum piece-square-table values for most pieces.

7. SUICIDE CHESS

The section 7 contains three subsections, viz. piece values (7.1), piece-square-table values (7.2), and a comparison of the learned values (7.3).

7.1 Piece Values

Figure 9 shows the development of the learned piece values for Suicide Chess. Note that all values are negative because we use the same chess engine for all chess variants, and this engine tries to maximise the material score. Usually a material advantage is advantageous, and for that reason, positive values are learned for all the pieces. In Suicide Chess, however, it is typically bad to have more pieces on the board than the opponent. Therefore, the pieces receive negative values, thereby leading to a high evaluation score if the player has *fewer* pieces on the board than the opponent.

Very low (very negative) values, such as the ones of the Rook and the Queen, signify that it is *bad* to have these pieces on the board and it is *good* if the opponent has these pieces on the board. The Rook's and Queen's values are due to their many move options. Anyone who has played Suicide Chess before is bound to have noticed how difficult it is to bar the Queen from attacking any of the opponent's pieces. The Rook, however, poses a bigger problem than the Queen, because the Queen usually has *multiple* options of attack. That leaves at least a few tactical options and the Queen can frequently quickly be moved towards her own demise. In contrast, it is easy for the opponent to position her pieces in a way, that enables the player's Rook to capture exactly *one* piece without afterwards being captured itself. Following this principle a player can often construct a critical chain of capturing moves. The Knight is barely more dangerous than the King because it is so easy to bereave the

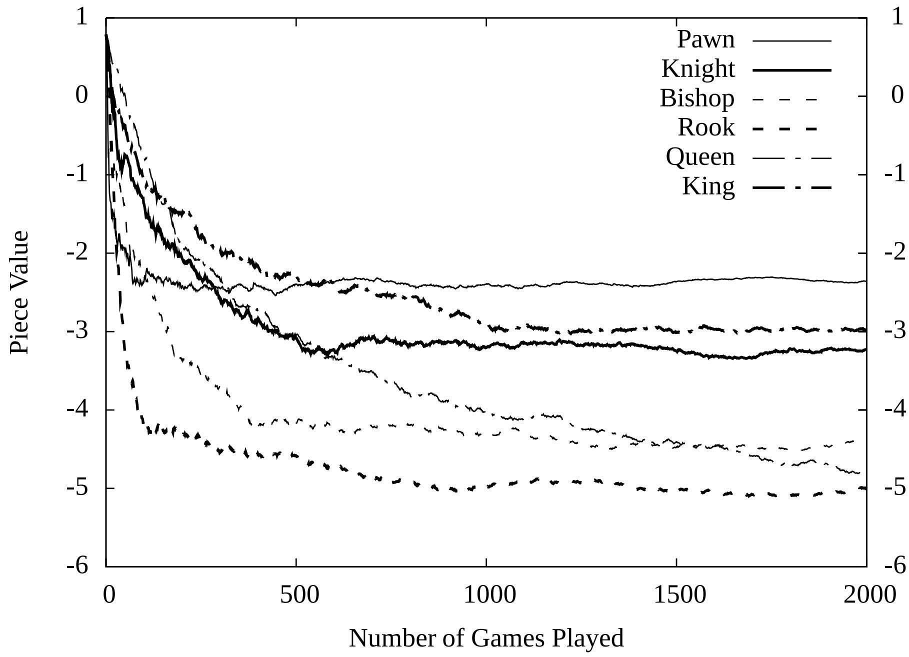


Figure 9: Development of material values (Suicide Chess).

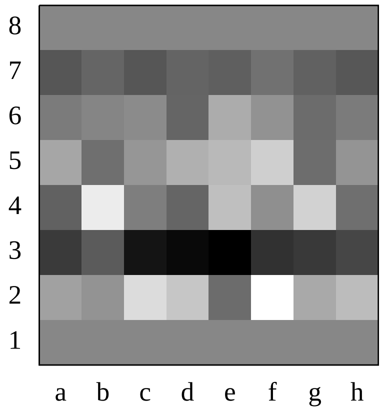
Knight of a majority of the move options by putting him close to the border of the board. Similarly, the Pawn's (absolute) low value can be explained by the limited capture options. Even if a Pawn ever captures a piece, it is almost impossible to create a capture chain as mentioned above with the Rook. Neither does a promotion pose any problem since Suicide Chess allows Pawns to be promoted to Kings.

The final values of the pieces are shown in Figure 11 together with their respective standard deviations. The normalisation here has been conducted to lead to a value of -5 for the Rook. This choice enabled a better comparability with the other values, which we will discuss further below (Table 5).

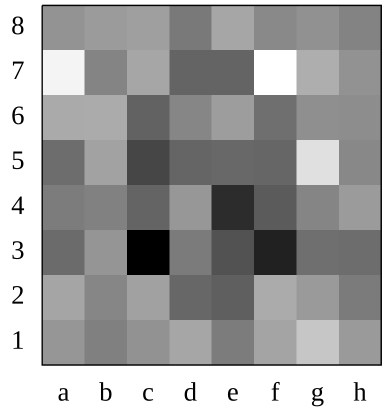
7.2 Piece-Square-Table Values

In Figure 10 the learned piece-square tables are shown for Suicide Chess. The diagrams 10.1–10.6 seem to be rather chaotic in comparison to the diagrams of the other chess variants. A certain symmetry is apparent in the diagrams of the Bishop and the Rook. But since the very bright and very dark squares are isolated and no gradients are visible the interpretation is rather troublesome. However, the quite big difference between the minimum and maximum values indicates that the differences are not due to chance fluctuations.

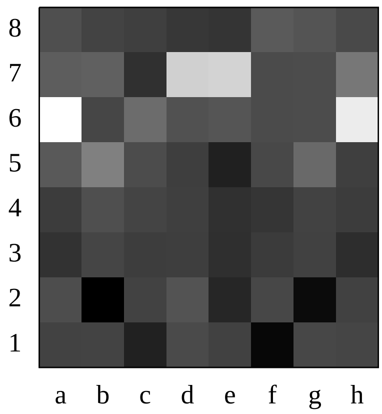
For Pawns one can, e.g., see that centre moves are not good openings moves. Moreover computer analysis in this game has shown that the opening moves h4, f4, Nf3, Nc3, d3, d4, e4, h3, b4, are all losing for White. With the exception of e4 and b4, these openings all have a negative piece-square-table value. However, one should be cautious with such an interpretation, as the learned values reflect the value of a square over the entire game, not only on the first move. Some other observations that can be made include, e.g., that Knights are strong on f7, but bad on c3 and f3, Bishops and Rooks are good on a6 and h6 (this is the square where they can be most quickly captured). The King has a much higher mobility, and is good on most squares, but it is strongly discouraged to move out of its original square, etc. Particularly interesting is the Queen's table, which shows that the Queen has only one particularly strong square, which is d7. In the opening, the Queen can be captured on this square by four pieces, and it will in addition open the opponent's Queen's file where possibly other pieces can be disposed.



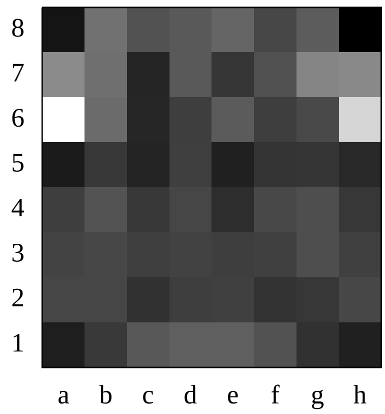
10.1: Pawn (Min/Max: -3.76/-2.45)



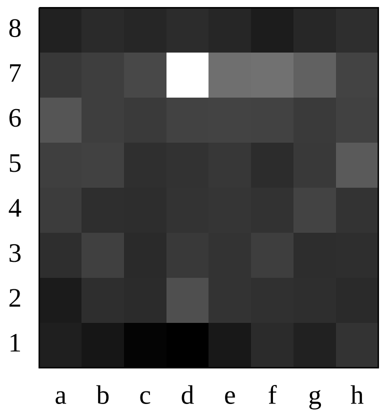
10.2: Knight (Min/Max: -4.09/-2.99)



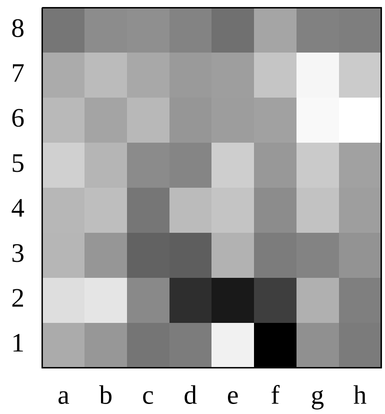
10.3: Bishop (Min/Max: -4.13/-1.64)



10.4: Rook (Min/Max: -5.43/-3.07)



10.5: Queen (Min/Max: -4.20/-1.64)



10.6: King (Min/Max: -3.15/-2.66)

Figure 10: Piece-square tables in Suicide Chess.

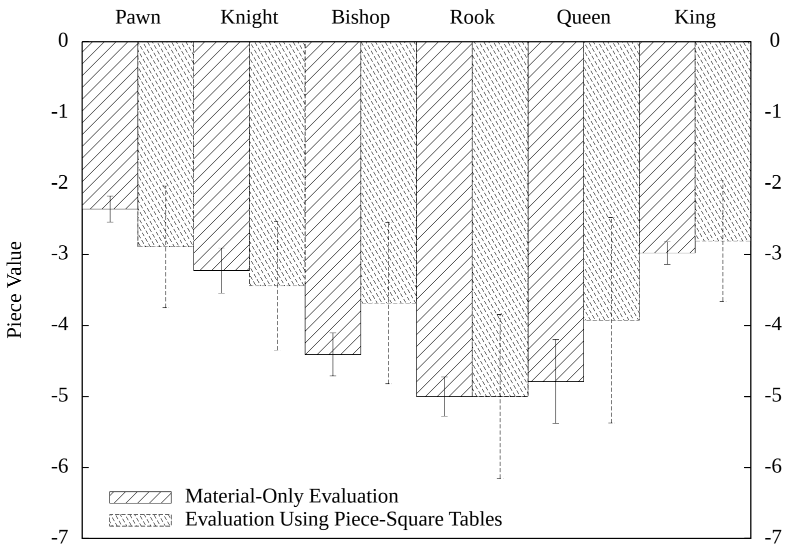


Figure 11: Final material values of the pieces (Suicide Chess).

7.3 Comparison of the Learned Values

Again, averaged values of the pieces have been calculated from the piece-square tables. The calculated values are listed in Table 5 together with the directly learned piece values. In their ordering, the values are quite consistent, but the values originating from the piece-square tables are somewhat lower relative to the Rook's values. The order is also quite consistent with general wisdom in this game, where the importance of the pieces is given as `R > K = Q = B > N > P`[13], meaning that Kings, Queens, and Bishops have approximately the same value, lower than Rook's and higher than Knights. Pawns have the lowest value. In our experiments, the King has received a much lower weight. It is unclear to us whether this is a significant deviation or whether this can be attributed to, e.g., the fact that the endgame has certainly had a lower importance in our experiments than in expert play.

	Piece-Square Values	Piece Values	DEEP SJENG (Losers)	DEEP SJENG (Suicide)
Pawn	-2.89	-2.36	1.14	0.50
Knight	-3.44	-3.22	4.57	5.00
Bishop	-3.68	-4.41	3.86	0.00
Rook	**-5.00**	**-5.00**	**5.00**	**5.00**
Queen	-3.92	-4.79	5.71	1.67
King	-2.81	-2.98	14.29	16.67

Table 5: Comparison of different sets of material values (Suicide Chess).

For comparison the table also includes values used by DEEP SJENG in two variants, Suicide Chess and Losers Chess.[14] The difference between the two is that in Losers Chess, the K king cannot be captured. What is surprising here, is the high value for the King. For the Losers Chess variant, we would not need any value for the King, and for the Suicide Chess variant the value of the King seems to be excessively high. The reason is probably that DEEP SJENG uses a very complex evaluation function, which is optimised for Suicide Chess, as opposed to the general one used in our experiments. Nevertheless, these values seem to be quite weak.

Results from the tournament are listed in Table 6. The entries in the first column have the following meanings.

learned : material-only evaluation function using the learned piece values

traditional : material-only evaluation function and inverted traditional material values, i.e., the values have been

[13] http://wiki.wildchess.org/wiki/index.php/Suicide
[14] These values can be found in the file `eval.c`

multiplied with -1. The King's value has been set to -2 because he moves and captures almost like a Pawn with some added mobility.

DEEP SJENG (Suicide) : material-only evaluation function and the material values used for Suicide by Deep Sjeng. All these values have been multiplied with -1 for adaption to the evaluation function used by DaSunsetter.

DEEP SJENG (Losers) : material-only evaluation evaluation function and the material values used for Losers by Deep Sjeng. All these values have been multiplied with -1 for adaption to the evaluation function used by DaSunsetter.

average p-sq : material-only evaluation function and the average piece-square values shown in Table 5

piece-square : the same evaluation function as above with the learnt piece-square values

Configuration	Won	Lost	Draw	Ratio
learned vs. traditional	1018	934	48	52.10%
learned vs. DEEP SJENG (Suicide)	1790	185	25	90.13%
learned vs. DEEP SJENG (Losers)	984	910	106	51.85%
learned vs. average p-sq	986	948	66	50.95%
piece-square vs. learned	1742	243	15	87.48%

Table 6: Results of the tournaments (Suicide Chess).

Despite the plausibility of the learned values, it seems that a straightforward inversion of traditional values yields almost the same results as the learned values. The win ratio is statistically significant with a p-value of 6% according to the conservative sign test, but we would nevertheless have expected a larger advantage here.

The bad performance of the suicide values of DEEP SJENG confirms the suspicion that DEEP SJENG uses for Suicide a strongly specialised evaluation function. The values taken from Losers Chess just perform better because of their similarity with the learned values. However, it shows that even substantial noise in the values (see Table 5) has only a slight impact on the results of the games.

Again, the engine with piece-square tables plays significantly better than the material-only version, which confirms that the learned piece values are not due to chance fluctuations. The exact placement of the pieces on the board seems to be of similar importance as in Standard Chess.

8. ATOMIC CHESS

Section 8 contains three subsections, viz. piece values (8.1), piece-square-table values (8.2) and a comparison of the learned values (8.3).

8.1 Piece Values

The normalisation of the Atomic Chess values has been conducted in a way that leads to a value of 2.5 for a Knight. The curves in Figure 12 show barely any changes after the 1000-th game. This means that the algorithm converges nicely for Atomic Chess and the 2,000 games conducted per run are absolutely sufficient.

Notice the extremely high value of the Queen which is hard to explain. The fact that (because of the explosion) every piece can only capture once per game leads to the assumption that with a skilled capture, a piece has to capture at least as many of the opponent's pieces as are needed to compensate for the loss of the capturing piece itself. However, with such a high value as the Queen's value in relation to the other pieces' values, this is barely achievable. Even if the Queen ($\approx$25 points) captures two enemy Rooks ($\approx$12 points), both Knights (5 points), and a Pawn[15] ($\approx$2,5 points) the capturing player still loses more points than his opponent. Therefore, the Queen's value is probably due to her high potential to threaten the King. Her high mobility and the potential of indirect check or checkmate give her the ability to threaten sometimes the King in two different ways[16] at the same time

[15] At most one Pawn can be captured at once since Pawns are not affected by the explosion.
[16] This is also possible in Standard Chess but in Atomic Chess it is much easier to achieve.

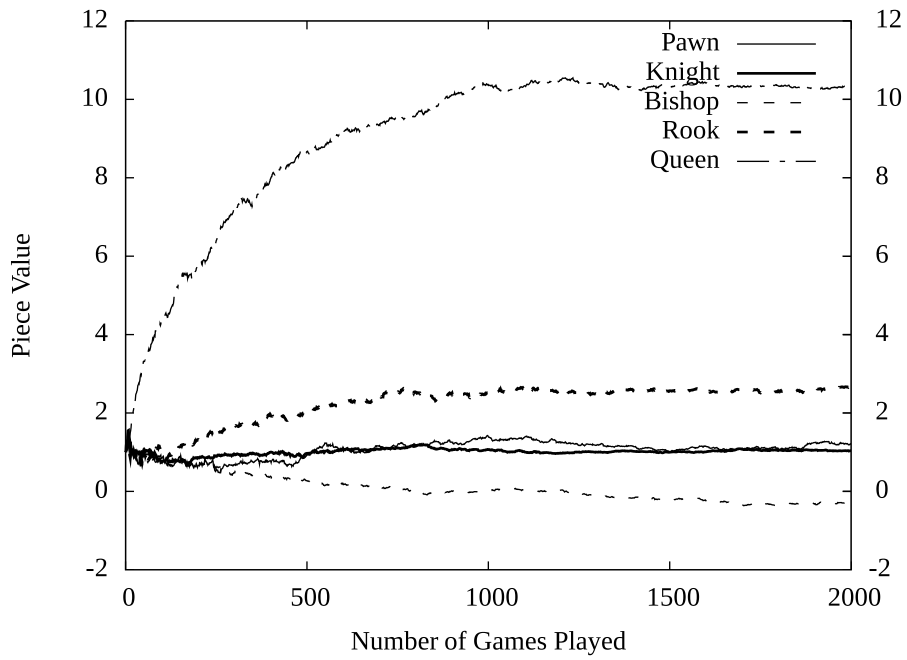


Figure 12: Development of material values (Atomic Chess).

and thereby checkmate him all by herself.

Somewhat surprising is also the slightly negative value of the Bishop. We do not have sufficient expertise for this game to confirm or disprove this statement. However, in a recent on-line book on Atomic Chess (Blackburn, www 2009), the Bishop does not play a prominent role. For example, the book contains separate chapters on tactics for the Queen, the Rook, and the Knight, but none for the Bishop. The Bishop is only discussed in certain endgames. An explanation for the negative value might be its unfortunate starting position. A threat to the Bishop on its starting square usually also threatens the Queen or the King (see Figure 13.1).

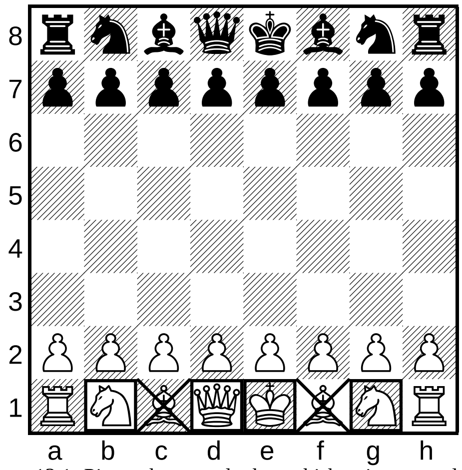
8.2 Piece-Square-Table Values

Figure 14.1 looks similar to its Standard Chess counterpart. A gradual transition from light at the bottom to dark at the top is apparent. Probably the reason for this is again the promotion of a Pawn upon reaching the eighth rank.

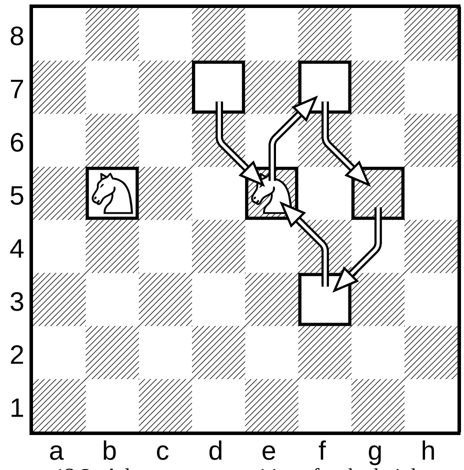
The Knight's diagram looks rather random at first, but a closer look reveals some patterns. Many of the slightly lighter patches are just a Knight's move apart (see Figure 13.2). In particular, the pattern reflects that 1.Nf3 is an extremely dangerous opening to which only f6 or e5 are playable to prevent both 2.Ne5 or 2.Ng5, both with imminent doom on f7. All these squares have very high values for the Knight. In contrast, its starting positions are particularly bad. This does not come as a surprise since a similar threat can appear here as shown in Section 7 by Figure 13.1.

In this context, the Bishop's diagram comes as a small surprise. In contrast to the results of directly learned piece values, the Bishop has a clearly positive value, with a small difference between the minimum (2.07) and the maximum (3.33) values. Also, the starting positions (c1 and f1) do not seem to be valued negatively. However, overall, the image looks rather random, and the low span between min and max values leads to the conclusion that bishop placements are not particularly important for the game.

The rook diagram clearly shows the danger of Rooks in this game. They can mate the opponent King from the distance by, e.g., moving over e1 to e8 or perform a similar manoeuvre to reach one of its neighbouring squares. The queen diagram is quite similar, but it has a much larger dark area, which probably results from the danger of capturing neighbouring pieces in case of an explosion if the Queen gets captured. However, even in this dark area, the Queen's value still exceeds the other pieces' values.



13.1: Pieces threatened when a bishop is captured



13.2: Advantageous positions for the knight

Figure 13: Diagram with the pieces on the board.

The reason for the upper area of the board being gray in the King's table (Figure 14.6) is again, that the King barely ever visited these squares. King safety is also a very important concept in Atomic Chess. Interestingly, the white and the black square in the diagram lie right next to each other: e1 is the safest square for the King, e2 the most dangerous one. The large difference between their (numerical) values illustrates that in Atomic Chess the King rarely ever moves in won games. In the experiment, this leads to the high value of square e1. If the game ever reaches a point where one player has to start moving his King around he usually loses the game. This leads to low or negative values for the squares d2, e2, f2 which are visited most often by an escaping King.

8.3 Comparison of Learned Values

Figure 15 shows calculated average values of the piece-square values with their respective standard deviations and the learned material values. Table 7 shows the final numeric values. However, we did not find any suitable comparison values for Atomic Chess.

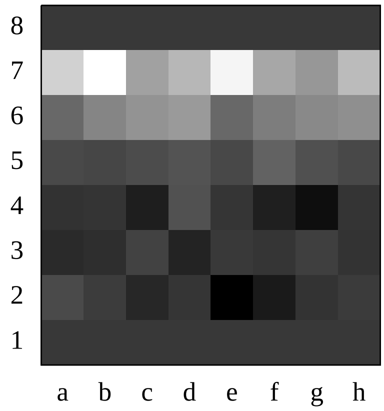
	Piece-Square Values	Piece Values
Pawn	1.95	2.91
Knight	**2.50**	**2.50**
Bishop	2.66	-0.77
Rook	5.19	6.37
Queen	13.98	24.77

Table 7: Calculated and learned material values in Atomic Chess.

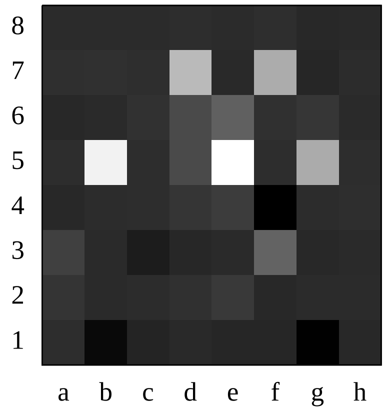
Configuration	Won	Lost	Draw	Ratio
learned vs. traditional	1231	753	16	61.95%
learned vs. average p-sq	1186	797	17	59.73%
piece-square vs. learned	1924	75	1	96.23%

Table 8: Results of the tournament (Atomic Chess).

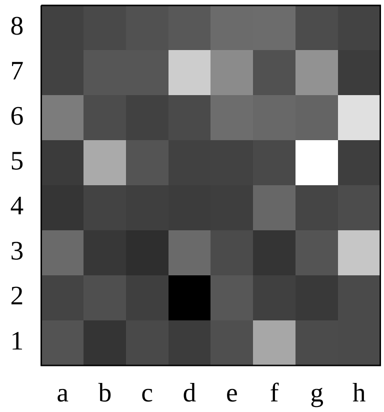
The results of the tournament shown in Table 8 lead to the same conclusion as the two previous sections. The learned values induce a winning ratio of 61.95% meaning stronger playing and a better evaluation function. In this experiment, directly learning the piece values was more successful than averaging their piece-square-table values. The key difference between these two versions is the much higher relative value of the Queen and the slightly negative value for the Bishop. The very good performance of the piece-square tables confirms the relatively high importance of the absolute positions of the pieces on the board.



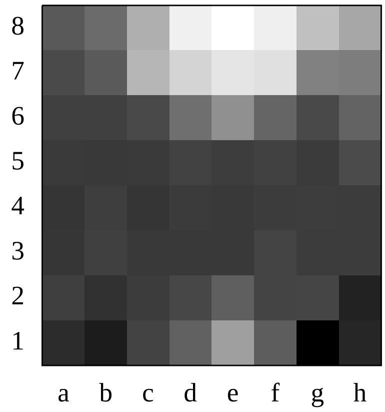
14.1: Pawn (Min/Max: 0.54/6.81)



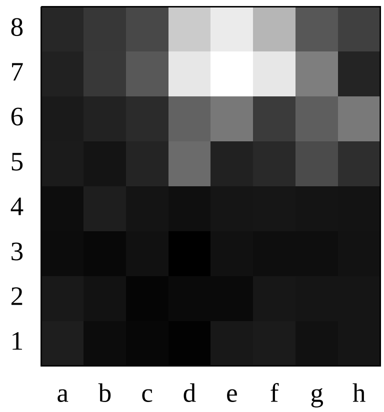
14.2: Knight (Min/Max: 1.74/5.67)



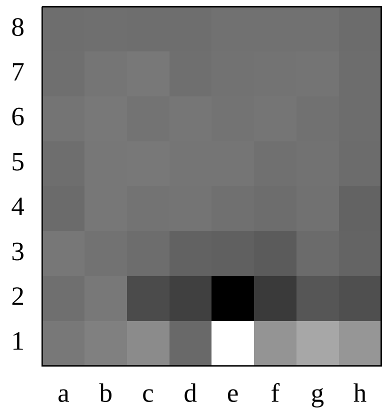
14.3: Bishop (Min/Max: 2.07/3.33)



14.4: Rook (Min/Max: 4.48/7.73)



14.5: Queen (Min/Max: 13.25/22.90)



14.6: King (Min/Max: -2.5/5.54)

Figure 14: Piece-square tables in Atomic Chess.

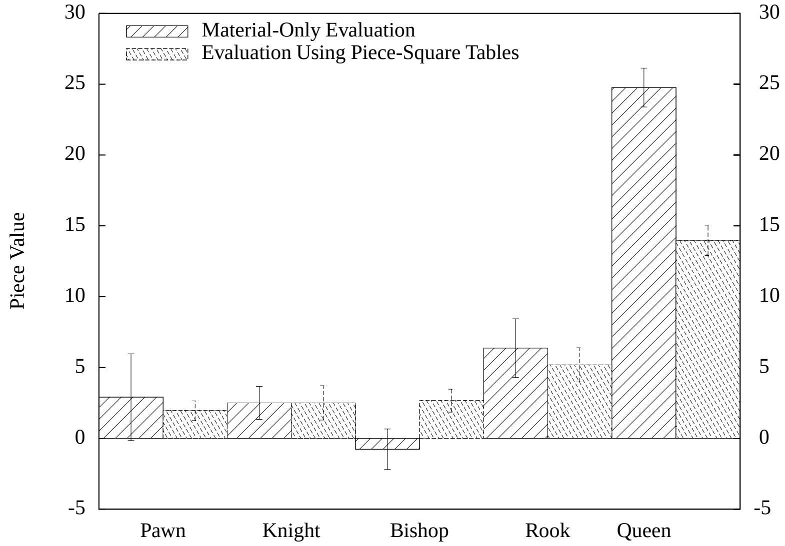


Figure 15: Final normalised material values of the pieces (Atomic Chess).

9. SUMMARY

The key result of our work are concrete values for the pieces of three popular chess variants, viz. Crazyhouse Chess, Suicide Chess, and Atomic Chess. Table 9 summarises the best-performing values for each variant. Here, we have normalised it in such a way that a Pawn has the (absolute) value 1. The first two lines contain values resulting from weighted averages of piece-square-table values, the last two have been directly learned.

Variant	Pawn	Knight	Bishop	Rook	Queen	King
Standard	1.0	2.7	2.9	4.3	8.9	∞
Crazyhouse	1.0	2.0	2.4	2.4	3.8	∞
Suicide	-1.0	-1.4	-1.9	-2.1	-2.0	-1.3
Atomic	1.0	0.9	0.0	2.2	8.5	∞

Table 9: Summary of the best-performing values, normalised to a Pawn.

These values have to be interpreted with a bit of caution in the sense that we expect that minor optimisations could still improve them. For example, in Suicide Chess, the values resulting from averaging the piece-square tables were a bit different for most pieces (except the Rook), but performed almost as well. In contrast Atomic Chess, the alternative version that did not have a 0 for the Bishop performed considerably worse. It is, nevertheless, quite counter-intuitive to attribute no importance to losing a Bishop (the weight was, in fact, even slightly negative), but we believe this deserves further investigation.

More importantly, we must keep in mind that the values have been obtained based on a material-only evaluation and it is, of course, not clear how these values will develop in the context of a fully fledged evaluation function. However, we are nevertheless quite confident that these values are reasonable, as can also be witnessed by the values we have obtained for Standard Chess, which are completely in line with standard chess theory (a Bishop is a bit more than a Knight, and both are typically a bit less than three Pawns, etc.).

We have also obtained piece-square tables for each of these variants, which show for each piece how valuable it is if placed on a particular square. Not surprisingly, in all variants, the piece-square tables clearly outperformed a material-only evaluation function. However, in the Crazyhouse Chess and Atomic Chess, where tactics play a much stronger role than in Standard Chess, this performance gain was much more pronounced (the material-only version won less than 4% of the games, as opposed to more than 12% for Suicide Chess and Standard Chess). In these games, the piece-value tables clearly showed the squares that are most likely to deliver a deadly attack to the uncastled King.

A possible point of criticism towards this approach is that the piece-square tables are static in the sense that they cannot adapt to new situations. For example, we have discussed that in Crazyhouse Chess, castling to the Queenside is rare, and that the piece-square values reflect this by completely neglecting the Queenside. This is good if we have an opponent that plays according to the same generally accepted principles, but it might be a major weakness if the opponent tries to exploit the program's weakness by a deliberately castling to the Queenside, not because it is an objectively good move, but simply because it knows that the program prefers Kingside squares, no matter what. This is a valid point, but one that goes beyond this study. To solve this issue, one would need something like piece-square values that are relative to the position of the two Kings. The problem can be found in regular chess as well (for a similar reason, e.g., chess programs use different piece-square values for the King in the middlegame and in the endgame), but it is somewhat more pronounced here, because the dynamics of the game depend strongly on the positions of the two Kings. It is an open question how to address this problem adequately.

A second point of research to be addressed in future work is to learn entire evaluation functions for three chess variants. For example, a small experiment was conducted at the end of the project where a new 14-weight evaluation function for Suicide Chess was trained. The idea behind this evaluation was a very primitive recognition of positions that contain the potential of capture chains. The engine with this special evaluation played 2000 games against an engine with learned piece-square tables and won almost 90% of those games.

However, in addition to the crucial question of the scalability of the techniques, the key problem here is to come up with useful features for an evaluation function. While research in computer chess has developed a large set of useful features that can be used in evaluation functions for Standard Chess, it is less clear what features should be used in the evaluation functions for these chess variants. In particular, static positional features will probably play a much smaller role in variants such as Crazyhouse Chess, which is dominated by tactics. In the ground-breaking work on his Checkers player, Samuel (1959) already proposed techniques for automatically tuning the evaluation which pioneered later work on reinforcement learning, such as the techniques that we used in this paper. He concluded his paper by making the point that the most promising road towards further improvements of his approach might be *"... to get the program to generate its own parameters for the evaluation polynomial"* instead of learning only weights for manually constructed features. Eight years later, in a follow-up paper, Samuel (1967) remarked that the goal of *"... getting the program to generate its own parameters remains as far in the future as it seemed to be in 1959."* Another forty years later, we still do not have much to add to that.

REFERENCES

Baxter, J., Tridgell, A., and Weaver, L. (1998a). A chess program that learns by combining TD(lambda) with game-tree search. In *Proceedings of the 15th International Conference on Machine Learning (ICML-98)*, pp. 28–36, Madison, WI, 1998. Morgan Kaufmann.

Baxter, J., Tridgell, A., and Weaver, L. (1998b) Experiments in parameter learning using temporal differences. *International Computer Chess Association Journal*, Vol. 21, No. 2, pp. 84–99. ISSN 1389-6911.

Baxter, J., Tridgell, A., and Weaver, L. (1998c). Tdleaf(lambda): Combining temporal difference learning with game-tree search. *Australian Journal of Intelligent Information Processing Systems*, Vol. 5, No. 1, pp. 39–43. ISSN 1321-2133.

Baxter, J., Tridgell, A., and Weaver, L. (2000). Learning to play chess using temporal differences. *Machine Learning*, Vol. 40, No. 3, pp. 243–263. ISSN 0885-6125.

Baxter, J., Tridgell, A., and Weaver, L. (2001). Reinforcement learning and chess. In Fürnkranz and Kubat (2001), chapter 5, pp. 91–116.

Beal, D. F., and Smith, M. C. (1997). Learning piece values using temporal differences. *International Computer Chess Association Journal*, Vol. 20, No. 3, pp. 147–151. ISSN 1389-6911.

Beal, D. F., and Smith, M. C. (1999a). Learning piece-square values using temporal differences. *International Computer Chess Association Journal*, Vol. 22, No. 4, pp. 223–235. ISSN 1389-6911.

Beal, D. F., and Smith, M. C. (1999b). Temporal coherence and prediction decay in TD-learning. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pp. 564–569. Morgan Kaufmann.

Beal, D. F., and Smith, M. C. (2001). Temporal difference learning applied to game playing and the results of application to Shogi. *Theoretical Computer Science*, Vol. 252, No. 1-2, pp. 105–119. Special Issue on Papers from the Computers and Games 1998 Conference. ISSN 0304-3975.

Beal, D. F., and Smith, M. C. (1998). First results from using temporal difference learning in Shogi. In H. J. van den Herik and H. Iida (eds.) *Proceedings of the First International Conference on Computers and Games (CG-98)*, pp. 113–125, Tsukuba, Japan. Springer-Verlag.

Blackburn, L. *The Atomic Chess Book*. http://virtual.parkland.edu/lblackburn/Atomic/atomic.html.

Breuker, D. M., Uiterwijk, J. W. H. M., and van den Herik, H. J. (1997). Information in transposition tables. In H. van den Herik and J. Uiterwijk (eds.) *Advances in Computer Chess 8*, pp. 199–211. Universiteit Maastricht. ISBN 9-062-16234-7.

Ekker, R.-J. (2003). *Reinforcement Learning and Games*. Master's thesis, RijksUniversiteit Groningen, Groningen, The Netherlands.

Fürnkranz, J. (2007). Recent advances in machine learning and game playing. *ÖGAI Journal*, Vol. 26, No. 2. Special Issue on Computer Game Playing.

Fürnkranz, J. (2001). Machine learning in games: A survey. In Fürnkranz and Kubat (2001), chapter 2, pp. 11–59.

Fürnkranz, J. (1996). Machine learning in computer chess: The next generation. *International Computer Chess Association Journal*, Vol. 19, No. 3, pp. 147–160. ISSN 1389-6911.

Fürnkranz, J. and Kubat, M. (2001). *Machines that Learn to Play Games*. Nova Science Publishers, Huntington, NY. ISBN 1-59033-021-8.

Gligoric, S. (2003). *Shall We Play Fischerandom Chess?* Batsford Chess Books. ISBN 0-713-48764-0.

Kaindl, H. (1982). Quiescence search in computer chess. *SIGART Newsletter*, 80, pp. 124–131. ISSN 0163-5719.

Knuth, D. E. and Moore, R. W. (1975). An analysis of alpha-beta pruning. *Artificial Intelligence*, Vol. 6, No. 4, pp. 293–326. ISSN 0004-3702.

Levy, D. N. and Newborn, M. (1991). *How Computers Play Chess*. Computer Science Press. ISBN 4-871-87801-2.

Pritchard, D. B. (2007). *The Classified Encyclopedia of Chess Variants*. John Beasley, 2nd revised edition. ISBN 0-955-51680-1.

van Reem, E. (2006). Shredder wins second Chess960 Computer Chess World Championship. *International Computer Games Association Journal*, Vol. 29, No. 4, pp. 214-218. ISSN 1389-6911.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, Vol. 3, No. 3, pp. 211–229. ISSN 0018-8646.

Samuel, A. L. (1967). Some studies in machine learning using the game of checkers. II — recent progress. *IBM Journal of Research and Development*, Vol. 11, No. 6, pp. 601–617. ISSN 0018-8646.

Skiena, S. S. (1986). An overview of machine learning in computer chess. *International Computer Chess Association Journal*, Vol. 9, No. 1, pp. 20–28. ISSN 1389-6911.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, Vol. 3, pp. 9–44. ISSN 0885-6125.

Sutton, R. S. and Barto, A. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA. ISBN 0-262-19398-6.

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, 8, pp. 257–278. ISSN 0885-6125.

Tesauro, G. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM*, Vol. 38, No. 3, pp. 58–68. ISSN 0001-0782.