An Investigation of Combinatory Mathematics' Influence On the Chess Games of Computer-Players

Matthew O'Connell

Computer Science

Mr. Pope

Abstract

This investigation analyzed combinatory mathematics' influence on the decisions of computer players with emphasize on the endgames of chess matches and with respect to chess' solvability.

Concepts of Computational Game Theory (CGT) such as zugzwang or "compulsion driven gameplay" are what allow computers to solve games (where "solving" is defined as determining who will win if both players play perfectly). A chess match can be solved once it has reached a point where the last player to move will be the loser. This is a scenario that was determined to occur most readily in chess endgames leaving computer players with one goal in order to win: reach the endgame to preform calculations using CGT.

To reach the endgame, computers use the minimax algorithm which provides a means to evaluate what move a computer player should make. Computer players' use of zugzwang and depths of tree searches were evaluated as an important facet of a computer player's programming because they often differentiate good players from bad players (where depths of tree searches convey how many moves into the future a program evaluates). The difference is due to more effective algorithms being able to better foresee zugzwang's influence as well as more moves into the future of a given chess match.

Finally the pruning stage, a means to optimize the minimax algorithm, was investigated. It takes into account the limitations of CGT imposed by Computational Complexity Theory, zugzwang, and the greater whole of CGT in order to increase the depth of minimax searches. It was determined to be the most influential decision-making algorithm of a computer chess program.

Combinatorial Game Theory's applications to chess continue to help us understand chess' insolvability and reveal better methodologies to tackle problems of the current chess age.

Table of Contents

Introduction: "Why chess remains an unsolved game"

Programmers have long tried to create unbeatable computer opponents that have the power to "solve" games. A program solves a game by determining exactly who will win given that both players play in their best interests (Allis 1994). For example, programmers have developed software that will always win at Connect Four. In fact, programmers around the world use their programs to participate annually in competitions such as the World Computer Chess Championship which has been held since 1974 (ICGA 2014). But despite the increasing skill of these computer programs, chess remains unsolved. This investigation will explain why via an analysis of the limitations and influence of Combinatorial Game Theory, a unique facet of Chess Theory and combinatory mathematics that leading software engineers still work to integrate into their programming.

CGT: "What makes chess an imperfect combinatorial game"

Combinatorial Game Theory (CGT) resides in a branch of theoretical mathematics that stems from the distinct classical game theory of economics (Eppstein 2014). While the latter is not included in the scope of this investigation, the distinction present between the two theories can be used to define the unique and intriguing aspects of CGT. A combinatorial game meets the following criterion:

1.	The game is played by two players.
2.	The game does not invoke any aspects of probability; instead, the outcome of the game is exclusively reliant on the players' abilities and the games nature.
3.	All of the game's information is displayed to both players.
4.	The game is turn-based.
5.	There is an absolute winner at the end of the game. This winner is determined as the player to move last.

Table 1. Aspects of a combinatorial game (Garner)

In layman's terms, a combinatorial game can be viewed by a computer as a combination. Using CGT, computers have been able to solve games like Connect Four because they represent simple combinations that computers can reason their way through. Chess, in reference to Table 1, easily satisfies combinatorial game constituents 1-4. However the fifth delineates that chess, because it can have a winner by checkmate or opponent's resignation, is not a perfect combinatorial game. Because chess is an imperfect combinatorial game, computers split chess matches in independent sub-games which *are* perfect. They can then analyze sub-games independently. However, computers are not able to solve a chess game at its inception.

Chess is limited to an 8X8 board wherein independent sub-games are not always easy to decompose (Elkies 1996) and independently compute. Figure 1 depicts a position from a chess match from the Zurich Chess Challenge Blitz between Hikaru Nakamura and Magnus Carlsen in

2014. Despite the pieces on the board being spaced far apart, the game is tied together due to some pieces being able to threaten far and multiple positions.

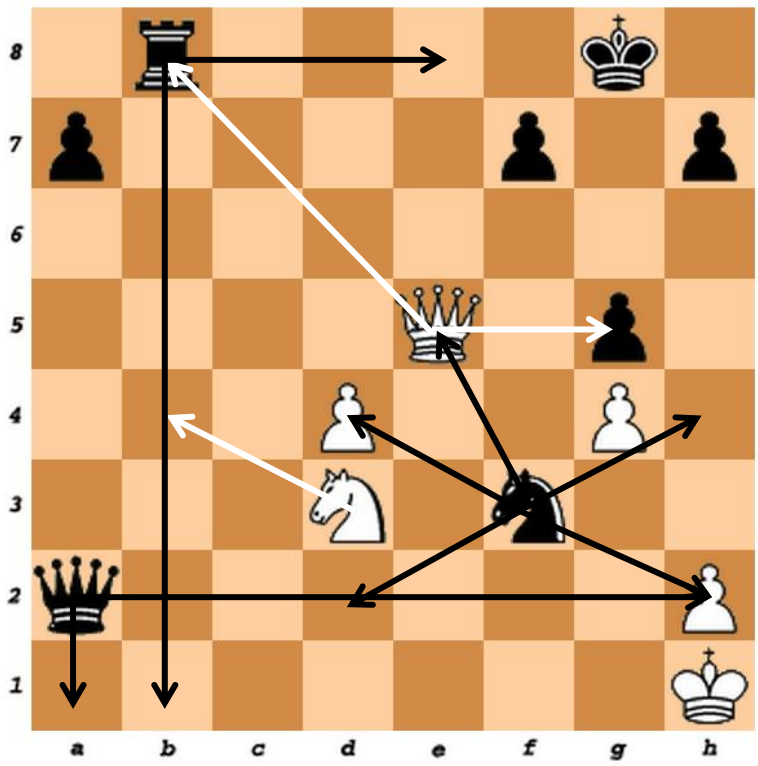

Figure 1. A position from a chess match where independent sub-games cannot yet be delineated. Arrows have been super-imposed to represent several threatened moves that tie potential sub-games into one. Graphic created by researcher.

Major pieces like the Queen have an influence that may extend to all corners of a board, tying different sub-games into one. However in the endgame of a chess match (at which point many of the greater pieces have been captured), the game transforms into complex problems rooted in CGT.

Zugzwang: "Compulsion Driven Gameplay"

In chess, players cannot choose to pass their turn. They are instead forced to take make a move, a rule that constitutes a fundamental concept of CGT known as zugzwang or "compulsion to move" ("Zugzwang" 2013). This common driving force of many endgames is what effectively crowns the player to move last, also known as the player to resist zugzwag the longest, as absolute winner of the game.

A common setup of this situation is depicted in Figure 2. This situation is referred to as a trebuchet and leaves both pawns in danger of attack from the enemy king. Each king guards their own pawn but now one must move. Given that it is the Black King's turn to move, it has only one other place on the board to move and still retain protection of the black pawn. This move would be to f5 however the White pawn prevents this move. Therefore the king must move elsewhere and its pawn will be captured. The white king is in the same situation with the square d4. Therefore this position holds mutual zugzwang; however because it is not White's turn, Black is forced to move. With Black's pawn unprotected, White easily seizes it and can freely progress its pawn to e8 where it can then be promoted to Queen and be used to win.

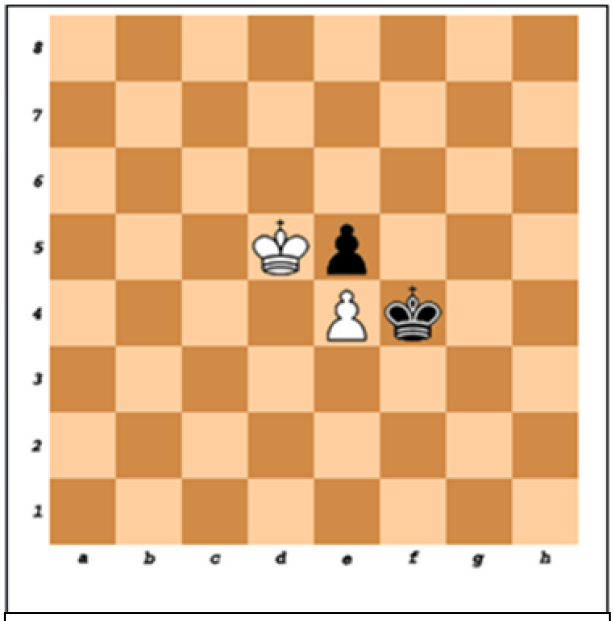

Figure 2. A Trebuchet. Graphic created by researcher.

The situation's complexity grows as more pieces are put into play. Figure 3's position was reached in a 1929 game between Schweda-Sika and Brno (Elkies 1996). A trebuchet ties the center four pieces together and promises a win to the player who can resist the center sub-games' zugzwang longest. Via CGT, the board can be decomposed into several distinct sub-games, namely the center chunk, the a & b file, and the h file. To determine the match's winner, each of these sub-games must be evaluated.

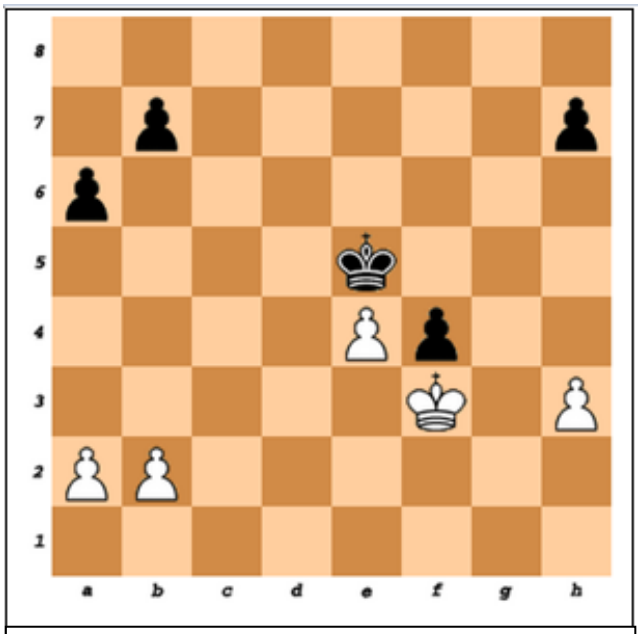

Figure 3. Schweda-Sika, Brno 1929. Graphic created by researcher.

The center chunk is simple: due to mutual zugzwang, neither king will move until they are forced to. Because the zugzwang is mutual, this sub-game can be considered a neutral position and ignored for now.

The h file contains two pawns which can only move closer to each other. However, because the black pawn resides on its initial square it can elect to move one *or* two spaces which grants it greater control of the subgame's zugzwang. For example, if the white h-file pawn was on h4, Black could move the black h-file pawn to h5 which will prevent the white pawn from moving further. This move would force White to then move the white king, losing the center chunk's pawn and then the match itself. Therefore the h-file sub-game reduces Black's zugzwang, giving Black a strong advantage.

The final sub-game to be evaluate is composed of the pawns residing within the a & b files. In this sub-game, both White and Black have pawns in their initial positions. However, one of Black's pieces is a space forward which prevents the advantageous option of a two-spaced move (analyzed earlier in the h-file sub-game). This limitation for Black's position gives White somewhat of an advantage within this sub-game. However because there are two pawns, the players have more options and the zugzwang is consequently mitigated. White's weak advantage from this sub-game is ultimately not enough to tilt the game in favor of White's control. Black will win if both players play perfectly; therefore, at this point the game is solved.

To summarize: Concepts of CGT such as zugzwang are what allow computers to solve games. A chess match can be solved once it has reached a point wherein zugzwang forces players to make a losing move. But until that point, chess matches remain unsolved. The final question is what prevents computers from solving games before that point? The answer lies in how a computer plays chess.

Game Complexity: "Computational Complexity Theory"

Just as every chess player is unique, so too is each chess computer. The algorithms of chess programs vary and are constantly being improved upon; but they all carry a fundamental weakness that prevents them from implementing their combinatory algorithms until a chess match's endgame. This weakness lies in game complexity, a facet of CGT's Computational Complexity Theory (CCT). A game's complexity is determined by the following five factors:

1.	State-Space Complexity: The number of legal game positions attainable from an initial game position.
2.	Game Tree Size: The number of possible games resulting from moves and the opponent's counters.
3.	Decision Complexity: The number of leaf nodes in the smallest decision tree.
4.	Game-Tree Complexity: The smallest number of positions to evaluate in a minimax [move-evaluation] search to determine initial position.
5.	Computational Complexity: The asymptotic difficulty of the game as it grows arbitrarily large.

Table 2. Factors of a Game Complexity (Allis 1994)

Computers play chess by using these values of game complexity. A rise in computer chess players has led to increasing research in CCT and its influence upon CGT. In 1950, Claude E. Shannon ascertained chess' Game Tree Size and State-Space Complexity (aspects 1 and 2 of Table 2) to be around the orders of 10^{43} and 10^{120} respectively (Shannon 1949). Victor Allis' studies from 1994 to modern day provide mathematicians with a smaller scoped estimate of Shannon's number and continue the search for better understanding (Allis 1994). However, these numbers remain extremely large and serve as the first hint as to why computers encounter difficulty in solving a full game of chess. Computers of the current chess era cannot compute all of these values (Shannon 1949); CCT conveys that CGT can only be effectively applied to the small sub-games of chess that stem from chess endgames. This leaves computer players' one goal: reach the endgame and then use CGT to win.

Chasing the Endgame: "How Do Computers Bring Opponents to the Endgame"

From the onset of a chess match, CGT makes it clear that a computer has one goal: reach the endgame. If a computer can reach the endgame, eliminating major pieces from the board, it can divide the game into sub-games. Then, by computing independent zugzwang values, it can make the match a perfect combinatorial game and solve it (Berlekamp n.d.). With this end goal in mind, the computer needs to proceed through the game evaluating how its moves will be countered.

In evaluating which move to make, the computer plays against itself several moves ahead and sees which will provide it with the best end scenario. Note: the resulting position is referred to as an end scenario and not an endgame. CCT explains that no computer can predict moves of an entire game (Shannon 1949) due to the vast number of possible moves. In fact, the number of moves ahead that a player can evaluate is a factor that divides good chess players and poor chess players both human and computer. The number of moves ahead that the computer evaluates is referred to as its depth. Figure 4 depicts a computer evaluating the beginning of a chess match to a depth of three moves.

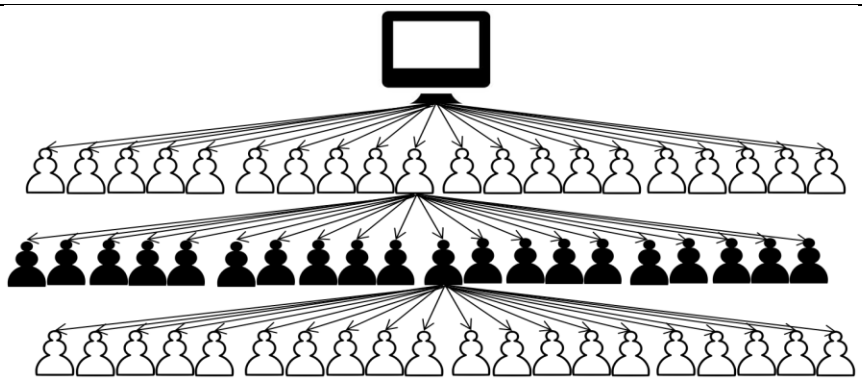

Figure 4. A visual depiction of a chess game's increasing complexity. Graphic created by researcher.

In the beginning of a chess match, White can play one of twenty moves. Black then plays one of twenty moves in response and then it is again White's turn. At this point, White may have much more than twenty moves to choose from but the state-space complexity of CCT explains that we already have a game-tree size of sixty possible games to evaluate with a number of total possible moves that approaches the asymptotic computational complexity. This algorithm of evaluating possible moves is referred to as the minimax search because it represents "an alternation between maximizing and minimizing the value of the positions in the tree" (Heilemann 2007). Algorithm 1 depicts a pseudo code interpretation of how a programmer might complete his or her program's minimax search algorithm.

```
int NegaMax (pos, depth)
{
    if (depth == 0) return Evaluate(pos);
    best = -INFINITY;
    succ = Successors(pos);
    while (not Empty(succ))
    {
        pos = RemoveOne(succ);
        value = -NegaMax(pos, depth-1);
        if (value > best) best = value;
    }
    return best;
}
```

- Evaluate() = an algoritm to evaluate if a given position is the player's "best interest"
- pos = The current position being evaluated
- best = a variable to hold the best choice
- depth = see page 10
- succ = decision complexity

Algorithm 1. C-based pseudo code depicting a typical minimax search (Heilemann 2007).

By examining this code it can be observed that the first variable to be evaluated and most influential variable in the program's expense is depth. Depth determines how many times the NegaMax algorithm will be recursively called within this method's while loop ultimately determining how long the program will take to make a decision. That decision is made based on which position is best as determined by an Evaluate method. The method gives positions integer

values that represent how preferable they are. Newly evaluated positions are compared against the current best within the last line of the while loop; and if the new position's value is better than the current best it becomes the best ("best = value").

A computer's lack of ability to evaluate every possible move in a chess match requires optimization of the minimax algorithm. Specifically, this optimization takes the form of limiting the number of moves evaluated to mitigate the chess match's game-tree complexity. A time-saving method to simplify a complex problem is known as a heuristic solution. This process is known as an Alpha-Beta search as well as, consistent with the tree analogy, "pruning" (depicted in Figure 5). Computers "chop off" branches of the game-tree leaving a much smaller number of scenarios to evaluate. In this example the game-tree size has halved.

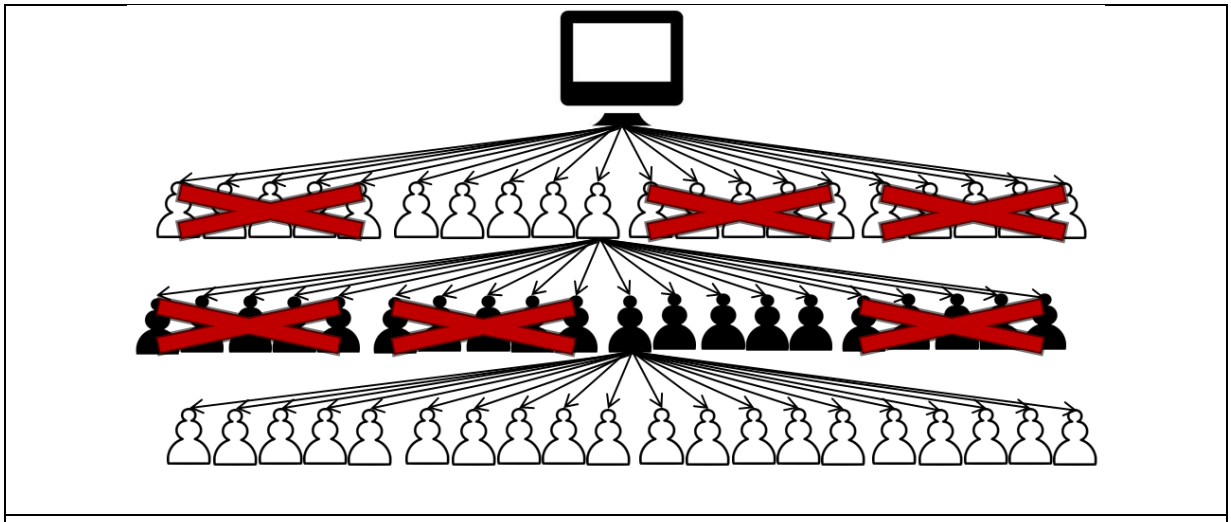

Figure 5. A visual depiction of a pruned chess game's complexity. Graphic created by researcher.

Computers choose which branches to prune based on a rating system that varies by the programmer's discretion. Because a computer will always assume its opponents will act in their "best interest", it prunes moves it considers poor. Combinatory Mathematics' zugzwang has a tremendous influence in this evaluation because of how difficult it is to determine which move is

in a player's "best interest". For example, a computer may believe that capturing a piece may be its best move when in reality zugzwang dictates that the computer should wait and instead force the opponent to capture in order to win.

```
int AlphaBeta (pos, depth, alpha, beta)
{
    if (depth == 0) return Evaluate(pos);
    best = -INFINITY;
    succ = Successors(pos);
    while (not Empty(succ) && best < beta)
    {
        pos = RemoveOne(succ);
        if (best > alpha) alpha = best;
        value = -AlphaBeta(pos, depth-1, -beta, -alpha);
        if (value > best) best = value;
    }
    return best;
}
```

- Evaluate() = an algoritm to evaluate if a given position is the player's "best interest"
- alpha & beta = lower and upper bounds of positions to evaluate
- pos = The current position being evaluated
- best = a variable to hold the best choice
- depth = see page 10
- succ = decision complexity

Algorithm 2. C-based pseudo code depicting a typical Alpha-Beta search or pruning process (Heilemann 2007).

The main difference between Algorithm 2 and Algorithm 1 is the inclusion of the beta and alpha variables. Acting as extra parameters in determining whether a position should be evaluated, they limit the amount of positions that a player computes. An extra AND operator in the while loop's parameters ("&& best < beta") restricts the loop to only look positions that are not preconceived as poor. This algorithm is dangerous because a computer can easily prune off its best moves from evaluation if the comparison is not well coded. Once again, a computer may give into zugzwang and try to capture a piece when it is better to wait.

In summary, the pruning stage (forced into use by a program's computational limitations) is the most influential decision-making algorithm of a computer chess program and must take into account CCT, zugzwang, and the greater whole of CGT in order to win.

Conclusions

This investigation began by evaluating chess' solvability by first delineating how chess is not a perfect combinatorial game. Chess' solvability was further diminished by the difficulty of sub-game delineation. With respect to the endgame, wherein it was determined that sub-games could be more easily delineated, the application of zugzwang, a fundamental principle of CGT, was evaluated in its role of solving sub-games. Then the algorithms that carry computer players' matches into those endgames were investigated along their limitations (as dictated by CCT). With reference to Shannon's findings and Allis' current work, it was determined that, without pruning, computers could not evaluate possibilities to great depth. Finally, the pruning algorithms, wherein zugzwang again played an instrumental role, demonstrated why computer chess players' skills vary by a programmer's opinion of what moves are in a player's "best interest". The subjectivity of this "best interest" evaluation proved to only further limit a computer's capability of solving chess. Within these pruning algorithms, a programmer's understanding of combinatory mathematics' CCT, CGT, and CGT's zugzwang, ultimately decided which computers could accommodate searches of the greatest depth and therefore preform best.

Chess masters continue to look for additional ways to outmatch and outthink their opponents. In these intense matches between highly skilled players, games often fall into endgame positions where resistance of zugzwang and control of the board dictate the winner. Combinatorial Game Theory remains a crucial aspect of these chess matches' endgames and continues to be studied by both chess players and mathematicians alike.

Furthermore, recent advances in artificial intelligence have prompted a new generation of chess grandmasters: computers. The programmers of chess-computers continue to compete around the world in pursuit of a computer with the ability to solve chess. Further analysis into Combinatorial Game Theory's applications to chess continue to reveal better methodologies to tackle problems of the hyper-modern chess age and pursue the goal of solving chess.

Bibliography

Allis, Victor E. Searching for Solutions in Games and Artificial Intelligence. PhD thesis, Department of Computer Science, University of Limburg, 1994.

Berlekamp, Elwyn. "Combinatorial Game Theory Background." *UCB Mathematics*. UC, Berkeley, n.d. Web. 14 June 2014. <http://math.berkeley.edu/~berlek/cgt/cgt-info.html>.

Elkies, Noam D. *On Numbers and Endgames: Combinatorial Game Theory in Chess Endgames*. Cambridge: MSRI, 1996. Print. Vol. 29 of *Games of No Chance*.

Eppstein, David. "Combinatorial Game Theory." *David Eppstein Research*. UC Irvine, n.d. Web. 14 June 2014. <http://www.ics.uci.edu/~eppstein/cgt/>.

Garner, Will. "A Brief Introduction to Combinatorial Game Theory." N.d. *Adobe Acrobat PDF* file.

Heilemann, Michael. "Chess Tree Search." *Computer Chess Programming*. 10 Apr. 2007. Web. 29 Sept. 2014.

ICGA. "ICGA Tournaments." *ICGA Tournaments*. International Computer Games Association, 28 Sept. 2014. Web. 28 Sept. 2014.

Shannon, Claude E. "Programming a Computer for Playing Chess." *Philosophical Magazine* 8 Nov. 1949: n. pag. Print.

"Zugzwang" Chess.com. Ed. Shaun Moves. Chess.com, 28 Aug. 2013. Web. 14 June 2014.