A New Chess Variant for Gaming AI

Azlan Iqbal

College of Information Technology, Universiti Tenaga Nasional, Kampus Putrajaya, Jalan IKRAM-UNITEN, 43000 Kajang, Selangor, Malaysia
azlan@uniten.edu.my

Abstract. In this article, we describe a newly-invented chess variant called Switch-Side Chain-Chess that is demonstrably more challenging for humans and computers than the standard, international version of the game. A new rule states that players have the choice to switch sides with each other if a continuous link of pieces is created on the board. This simple rule increases significantly the complexity of chess, as perceived by the players, but not the actual size of its game tree. The new variant therefore more easily allows board game researchers to focus on the 'higher level' aspects of intelligence such as perception and intuition without being constrained by a larger search space as they would be if using a game like Go or Arimaa. They can also immediately build upon the tried and tested approaches already being used in strong chess engines instead of having to start from scratch or a lower level of progress as is the case with other games of this type.

Keywords: Chess, variant, complexity, intelligence, intuition, perception.

1 Introduction

Chess variants may vary from the standard game in terms of factors like the types of pieces used, the shape and size of the board, and particularly the rules [1, 2]. The following provides a succinct description.

> "Chess variants comprise a family of strategy board games that are related to, inspired by, or similar enough to the game we today call Chess. The game we commonly know today was based on earlier games, most immediately the Arabian game of Shatranj, itself descended from the Indian game of Chaturanga. Besides its direct ancestors, Chess has many cousins, the most popular being Shogi (in Japan), Xiangqi (in China), and Janggi (in Korea). The modern game of Chess has also inspired countless variants. Some have been created by Chess champions seeking new challenges. Some have been created by entrepreneurs who have provided commercial sets. Some have been created for fairy Chess problems without any intent of actually playing them. And most have been designed by creative people who like to try out new pieces, new rules, or new ideas." [2]

J.C. Anacleto et al. (Eds.): ICEC 2013, LNCS 8215, pp. 9–16, 2013.

Our purpose was to challenge the field of artificial intelligence (AI) with a new Drosophila [3]. One of the problems with standard chess AI today is that computers are already able to play it well enough using efficient search techniques and well-designed heuristics. This has lead to the intensified study of board games with greater search spaces (i.e. the size of the game tree) such as Go and Arimaa in the hope that designing computer programs to play them as well as the best human players will lead to more advances in AI [4, 5]. Even so, many of the techniques that have worked for chess seem to also work well for these games [4]. Research into chess and computer chess has, in fact, yielded many benefits related to various fields such as molecular computing [6, 7], automated theorem proving [8], computer music composition [9], machine reading [10], cognitive development [11], the education of children [12, 13] and medicine, specifically with regard to Alzheimer's disease [14-16].

It is therefore not inconceivable that the right chess variant could be just as beneficial, if not more; and not only to AI. The advantage of our proposed variant, Switch-Side Chain-Chess (SSCC)[1] over more complex games and other chess variants is that it increases the human-perceived complexity without increasing the size of the original game's tree. This encourages the development of more sophisticated AI without the computational burden of a greater search space. In the following section, we explain the new rule of SSCC in some detail. In section 3, we discuss and illustrate with examples the impact and significance of the new variant in contrast to the standard game. We conclude in section 4 with some suggestions for further work.

2 The New Rule

In SSCC, the additional rule is simply that when a 'chain' or a link of pieces is formed on the board by the piece that moved last – enclosing at least two empty adjacent squares – the player has the right to switch armies with his opponent. Fig. 1 shows example configurations of chains. In (a), taken from a real game, the white knight has just moved from the *g6* square to *e7*, delivering check on the enemy king. The knight creates the chain *g5-h6-h7-g8-f7-e7-f6-g5* and now the player in control of the white army has the right to switch sides with the opponent, if so desired. The turn is then Black's, regardless of whom is now in control of that army. In this particular case, there is a *double* check created and therefore the enemy king *must* move.

Since the king is unable to create a new chain in the process in order to switch back, it may not make much sense for White to switch in the first place. In (b), a constructed position, it is shown how two empty squares diagonally adjacent may constitute a valid chain as well. Logically, with fewer than 6 pieces on the board, SSCC reverts back into the standard version's endgame. We could not find this particular concept of a chain in any existing variant. The simple addition of a new rule in SSCC introduces levels of complexity beyond the standard game without

[1] Iqbal, M. A. M. Apparatus for Playing a Chess Variant and Its Method. Malaysia Patent Application No. PI 2011006257. Filing Date: 23 December 2011.

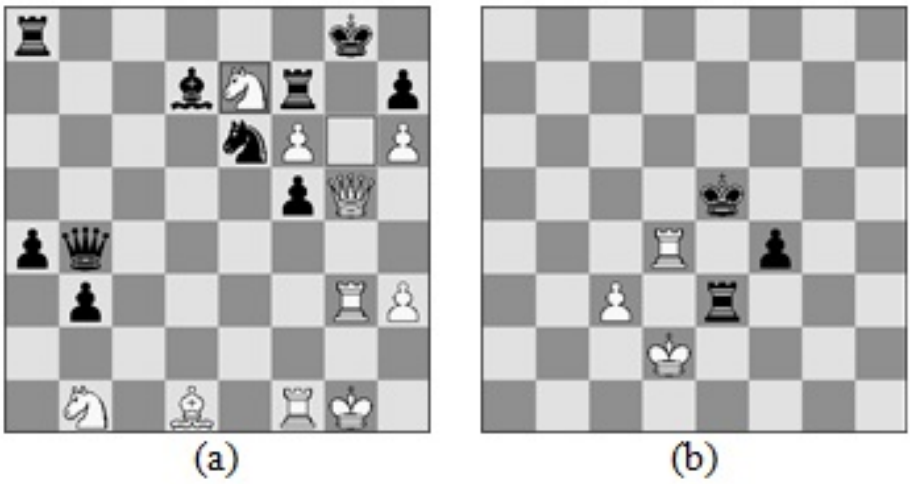

Fig. 1. Chains in the new variant

affecting the size of the original game's tree. It can be played on any standard chess set but a board that allows for easy rotation – or used in conjunction with a turntable device – would be preferable.

3 Increased Human-Perceived Complexity

If a computer program could play SSCC *well*, it would first need to include all the heuristics that work for standard chess (every chess game is an SSCC game) but also additional heuristics that deal with the complexities of 1) chain detection on the board, and 2) deciding whether or not to switch sides. First let us consider the standard game's tree and see why switching sides does not affect it. Consider the game tree of tic-tac-toe shown in Fig. 2[2] as a small-scale example.

If we consider 'X' to be the *player* in control of the white pieces and '0' to be the *player* in control of the black pieces, at any point, if a chain formed on the board and the players switched sides, the *tree itself* would be unaffected; i.e. no additional positions or nodes would be created. The switch is analogous to the players simply exchanging seats with each other. While this does not affect the 'physical' game tree, the players themselves are faced with additional challenges. It is interesting to note that, computationally, an implementation using a *larger* game tree is possible – e.g. by mapping player identity to piece color or treating the decision to switch as an extra 'move' – but this is not a necessity as the standard 'minimax' decision rule usually employed can be simply inverted at the appropriate time. This is the reason why a single computer chess engine can easily play a fair and unbiased game against itself.

[2] Adapted from: `http://ozark.hendrix.edu/~burch/cs/150/test/fr/print.html`

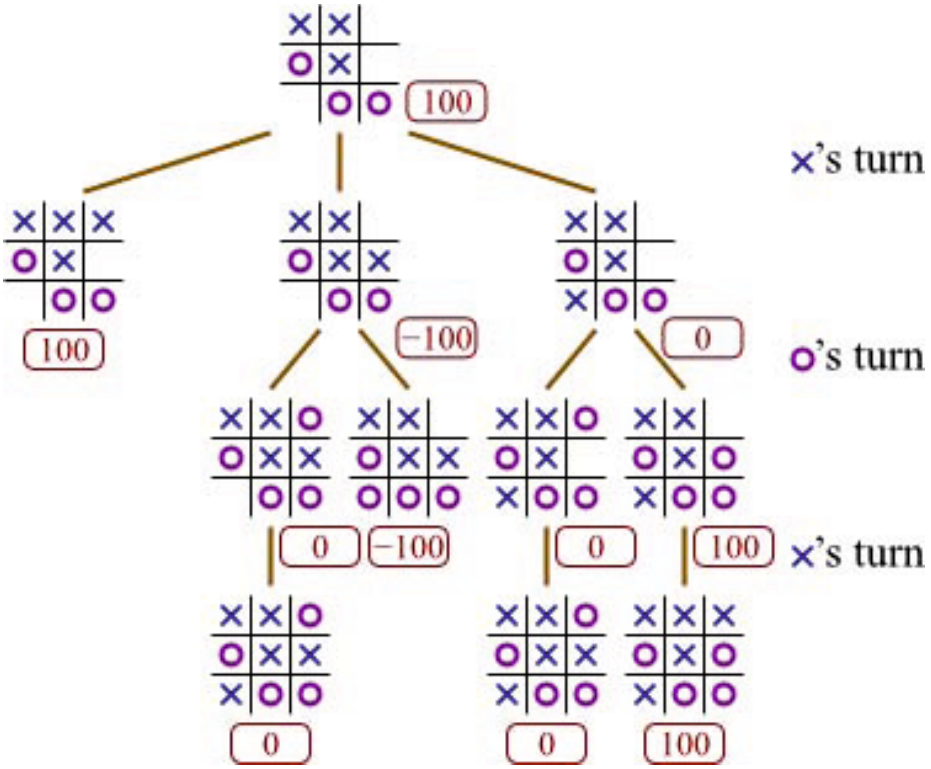

Fig. 2. The typical game tree structure

Experienced SSCC players, for instance (there are none at present), might find the variant easier than standard chess because they are particularly good at pattern recognition and exploiting opportunities on the board related to switching sides. Experienced standard chess players might find it much more difficult because they are simply not used to the new rule. Therefore, at this time, we can rely only on the logical argument that SSCC demands more from players. Computationally, the difficulties become more evident. A computer program requires heuristics that make chain detection both thorough and efficient. Even with knowledge of graph theory, there is no easy way for a computer to 'see' a SSCC chain (they can take many forms on the board) without repeatedly examining virtually every piece in a position and its surrounding squares since every piece is potentially the starting and ending vertex.

If one extends this concept to the millions of positions analyzed at every move, chain detection alone becomes too resource intensive. Here lies the first main challenge to gaming AI that will not lend itself easily to a brute-force solution. It may be acceptable for a human to miss certain chains but a computer must always be 'aware' of all possible chains to prevent the human player from

cheating, or simply being able to proceed legally with a switch. Let us say that this first main challenge is met. The computer is then faced with the second main challenge in having to decide whether or not to switch sides. Earlier, it was mentioned that the minimax decision rule used in standard chess can be inverted should a chain be formed.

This means that, should a candidate move by the computer create a chain (and switching becomes possible), the computer should now consider the perspective of the opponent since it could assume immediate control of his army. But is this sufficient to play SSCC well? We developed a rudimentary prototype SSCC program using this inverted minimax or 'iminimax' decision rule and tested it informally against a few average chess players. Due to an inability to detect chains correctly and efficiently enough, the program played at a very poor level. However, even if it could detect chains well, what sort of switching heuristics would bring it to the 'world class' level? Consider the position shown in Fig. 3 which was taken from a real SSCC game between the author and his research assistant.

Why is Bf7 a critical mistake when, in standard chess, it looks perfectly sound compared to the Kd8 alternative? The following analysis – which was performed after the game – perhaps illustrates how complicated SSCC can get. The reader, with the aid of a chessboard and the information in section 2, might like to determine the best sequence of moves before looking at the analysis. '(S)' indicates the presence of a chain and the decision to switch. Where the chain is not easily apparent, it is included.

> **1. Qh5+ Bf7? 2. Qh6** (S) **Bh5** (S, this also removes the bishop as a defensive piece along the line and sets it up for capture by the queen) **3. Nxg5** (S, probably the only move that creates a valid chain in this position) **Qd8** (S, removing the d8 escape square of the king) **4. h4** (S, again probably the only chain move) **Nd7** (S, removes the last escape square) **5. Qxh5#**
> Alternatively, if **3. . . . Qd8** was missed and **3. . . . Qg4** played instead, 4. f4 (S, f4-g4-h5-h6-h7-g8-f8-e7-f6-g5-f4) Nd7 (S) **5. Bh3** (S, h3-g4-h5-h6-h7-g8-f8-e7-f6-g5-f4-g3-h3; **5. Bf3** also works) **Rd8** (S) **6. Bxg4** (S) **c6** (S) **7. Qxh5#** (*5. Bxh5# also works and creates another chain but it is unnecessary at this point*)

The basic idea here is that once a player is able to switch and gains control of the opponent's army, he can 'set him up' to lose as long as a sequence of chains can be guaranteed to lead to checkmate. It should be clear from the above example that this is not always easy to do. What combination of new heuristics and associated 'weights' would work well in SSCC? The act of switching sides goes completely against the progressive build-up of material (usually measured in 'pawn units') that chess engines generally rely on. When should the computer 'gamble' all that has been gained, on a switch? Human players would typically use intuition and other subtle factors in conjunction with their pattern-recognition abilities to decide. In short, SSCC appears to combine the zero-sum

Fig. 3. *Qh5+* and black replies with the critical mistake of *Bf7*. Why?

perfect-information nature of standard chess with some amount of chance or unpredictability [17] (will there be an opportunity to switch back somewhere down the line?) without affecting the 'certainty' of the standard game's tree.

4 Conclusions

The new variant proposed poses new, interesting challenges to gaming AI without the computational burden of a larger search space. Concepts such as 'intuition' and 'perception' which are critical to high-level play (when it comes to humans) are definitely worthy of investigation in the *computational* domain. Further work in the area would benefit from a creative, mathematical demonstration of the increased strategic complexity of SSCC, as we have presently only 'logically' argued for it. This will not be trivial if it is to be sufficiently convincing. It may be analogous in difficulty to a 'proof' of the estimated maximum number of forced three-move mate sequences possible in standard chess.

Further work should also include some new, cleverly-designed heuristics related to chain detection and switching; their effectiveness demonstrated using prototype SSCC game engines. Eventually, these technologies would do well if they could be extended to other areas of research such as image processing and intelligent real-world systems where decisions need to be made relatively quickly with only limited information available.

Acknowledgements. We thank Boshra Talebi Haghighi and Sinan Qahtan Mohammad Salih for their contributions. This research was sponsored in part by the Ministry of Higher Education (MOHE) in Malaysia under their Fundamental Research Grant Scheme (FRGS/1/10/TK/UNITEN/02/2) and the Ministry of Science, Technology and Innovation (MOSTI) in Malaysia under their eScienceFund research grant (01-02-03-SF0240).

References

1. Gollon, J.: Chess Variations: Ancient, Regional, and Modern. Charles E. Tuttle Co. Inc., North Clarendon (1973)
2. Bodlaender, H.L., Howe, D.: The Chess Variant Pages (2002), `http://www.chessvariants.com`
3. McCarthy, J.: AI as Sport. Science 276, 1518–1519 (1997)
4. Hsu, F.-H.: Cracking Go. IEEE Spectrum 44(10), 50–55 (2007)
5. Wu, D.J.: Move Ranking and Evaluation in the Game of Arimaa. BA diss. Harvard College, Cambridge, MA (2011)
6. Cukras, A.R., Faulhammer, D., Lipton, R.J., Landweber, L.F.: Chess Games: A Model for RNA Based Computation. Biosystems 52(1-3), 35–45 (1999)
7. Faulhammer, D., Cukras, A.R., Lipton, R.J., Landweber, L.F.: Molecular Computation: RNA Solutions to Chess Problems. PNAS 97(4), 1385–1389 (2000)
8. Newborn, M.: Deep Blue's Contribution to AI. Annals of Mathematics and Artificial Intelligence 28, 27–30 (2000)

9. Friedel, F.: Ludwig - A Synthesis of Chess and Music. Chess Base News (December 4, 2006), http://www.chessbase.com/newsprint.asp?newsid=3522
10. Etzioni, O., Banko, M., Cafarella, M.J.: Machine Reading. AAAI Spring Symposium on Machine Reading. Technical Report SS-07-06, 1-5 (2007)
11. O'Neil, H.F., Perez, R.: Computer Games and Team and Individual Learning. Elsevier (2007) ISBN: 0080453430
12. Ferreira, D., Palhares, P.: Chess and Problem Solving Involving Patterns. TMME 5(2 & 3), 249–256 (2008)
13. AF4C: First Move Program Info. America's Foundation for Chess (2008), http://af4c.memfirstclubs.net/club/scripts/section/section.asp?NS=FMPI
14. Cavezian, C., Berquand-Merle, M., Franck, N., Demily, C.: Self-training for Cognitive Remediation in Schizophrenia. Schizophrenia Research 98(suppl. 1), 53 (2008)
15. Ciamarra, M.: Checkmating Alzheimer's Disease. FIDE (2013), http://www.fide.com/component/content/article/1-fide-news/7066-checkmating-alzheimers-disease.html
16. Dahl, M.: Being a Bookworm Boosts Your Brainpower into Old Age. Today Com. (July 3, 2013), http://www.today.com/health/being-bookworm-boosts-your-brainpower-old-age-6C10532642
17. Rubin, J., Watson, I.: Computer Poker: A Review. Artificial Intelligence 175(5-6), 958–987 (2011)