Novel AI Strategies for *Multi*-Player Games at Intermediate Board States

Spencer Polk and B. John Oommen

School of Computer Science, Carleton University, Ottawa, Canada
andrewpolk@cmail.carleton.ca, oommen@scs.carleton.ca**

Abstract. This paper considers the problem of designing efficient AI strategies for playing games at intermediate board states. While general heuristic-based methods are applicable for *all* boards states, the search required in an alpha-beta scheme depends heavily on the move ordering. Determining the best move ordering to be used in the search is particularly interesting and complex in an intermediate board state, compared to the situation where the game starts with an initial board state, as we do not assume the availability of "Opening book" moves. Furthermore, unlike the *two*-player scenario that is traditionally analyzed, we investigate the more complex scenario when the game is a *multi*-player game, like Chinese Checkers. One recent approach, the Best-Reply Search (BRS), resolves this by a process of grouping opponents, which although successful, incurs a very large branching factor. To address this, the authors of this work earlier proposed the Threat-ADS move ordering heuristic, by augmenting the BRS by invoking techniques from the field of Adaptive Data Structures (ADSs) to order the moves. Indeed, the Threat-ADS performs well under a variety of parameters when the game was analyzed at or near the game's initial state. This work demonstrates that the Threat-ADS also serves as a solution to the unresolved question of finding a viable solution in the far-more variable, intermediate game states. Our present results confirm that the Threat-ADS performs well in these intermediate states for various games. Surprisingly, it, in fact, performs better in some cases, when compared to the start of the game.

1 Introduction

AI techniques have been used extensively to play games, and well-known approaches such as alpha-beta search are known to gain improved efficiency and performance through strong move ordering, or attempts to search the best move first [1]. While these techniques are generally applicable for *all* board positions, their applicability for intermediate positions is far from obvious. This is because, in investigating the performance of move ordering techniques, rather than considering random game positions or the initial board state, analysis at *intermediate*

** *Chancellor's Professor*; *Fellow: IEEE* and *Fellow: IAPR*. The second author is also an *Adjunct Professor* with the Dept. of ICT, University of Agder, Grimstad, Norway.

board states is particularly fascinating. This is because the number of possible positions and the consequent potential moves, in an intermediate position is far more than the number encountered at the starting position. This is the focus of this paper. However, rather than considering move ordering-based AI search strategies for the classical family of *two*-person games, we consider here the scenario when the game is a *multi*-player game. It is well-established that the research domain associated with Multi-Player Game Playing (MPGP) of adversarial games is relatively unexplored[1].

The majority of techniques utilized by MPGP, such as the Paranoid and Max-N algorithms, are extensions of well-understood, powerful two-player techniques, such as Alpha-Beta Search, to a multi-player environment [5, 6]. However, while these techniques are natural and make intuitive sense, for a variety of issues, they have trouble performing at a level comparable with their two-player counterparts, necessitating the development of new techniques, and improvements to existing ones [4, 7]. One very recent strategy, attempting to overcome weaknesses in tree pruning and processing time in multi-player environments, is the Best-Reply Search (BRS), which has been found to outperform its competitors in a variety of environments [8]. Despite its power, however, the nature of the BRS leads to a large branching factor in the game tree, thus making tree pruning an important area of consideration in its use [8].

The authors of this work proposed in [9] the Threat-ADS heuristic, which makes use of techniques from the formerly unrelated field of Adaptive Data Structures (ADS) to improve move ordering in the context of the BRS. The Threat-ADS heuristic was found to produce meaningful improvements in a variety of cases [9, 10] when the game started from its initial board position. This paper demonstrates that the Threat-ADS is also a viable solution to efficiently resolve the search problem for MPGP analyzed from intermediate states.

The remainder of the paper is laid out as follows. After a motivating section, Section 3 briefly discusses existing techniques for MPGP including the Threat-ADS. Section 4 describes the game models that we employ and specifies the experiments we have performed. Section 5 presents our results. Finally, Sections 6 and 7 contain our analysis and conclusions, and open avenues of further research.

2 Motivation

Investigation into AI-based game playing from intermediate board states is an interesting topic, as it involves a much richer area of inquiry than the starting position of the game. This is because:

[1] Recent years have seen a rising focus in the literature on this domain, accelerated, in part, by the growing popularity of multi-player board games, such as the highly-regarded Settlers of Catan, and the multi-player nature of electronic games [2, 3]. Despite increasing interest in both the academic and public domains, MPGP remains a relatively unexplored field, and significant work remains to be done before knowledge of MPGP approaches the vast body of work dedicated to adversarial two-player game playing [1, 2, 4].

1. “Opening book” moves are not available at an intermediate board state [11].
2. In all non-trivial games, the variability in possible intermediate board states is almost always exponentially large.
3. Unlike two-player games, in multi-player games, the number of opponents who pose an adversarial threat in an intermediate board state could be larger than the number that the perspective player faces at the start of the game.
4. In some cases, at an intermediate board state, the number of adversarial opponents could have reduced, due to player eliminations.
5. Apart from the number of opponents, the identity of the opponents that could potentially threaten the perspective player could change in the intermediate board sate.

To explain the above issues, it is clear that if a game is well known, a game playing engine will, generally, make use of an opening book of strong, known moves that are derived from either expert knowledge, or trained from previous experience with the game [11]. In the case of intermediate game states, however, such well-known opening moves are unavailable, and thus the efficiency of the game search is more important. Furthermore, the opening stages of the game have a relatively small degree of variability, compared to the vast number of possible intermediate board states. Thus, when analyzing the performance of move ordering, addressing intermediate board states provides much more powerful results related to its performance.

The problem is accentuated in the case of MPGP. In the case of a multi-player game, there are other considerations at work aside from the variability of the intermediate board states. It is typical that in a multi-player game, the winner is determined by the last player remaining in the game, thus over time it is possible that players will be eliminated from play. Considering that the Threat-ADS makes gains by prioritizing the most threatening player, it is possible that the removal of players from contention will have an impact on its performance. Furthermore, depending on the qualities of the game, different opponents may be more threatening at different stages of play. For example, if one opponent has performed very well thus far, he may be more capable of minimizing the perspective player’s score and pose a greater threat than he posed earlier in the game. Figure 1 shows an example of these effects in the context of the Virus Game [9, 10], explained below. An alternate example when the number of adversarial opponents increases, e.g., in Chinese Checkers, is easily derived.

Our goal is to find any expedient solution to the search problem when one starts from these intermediate positions. Of course, while one can use the established BRS to tackle the problem, we unarguably demonstrate that by augmenting it with an ADS, the previously-proposed Threat-ADS is capable of adapting to these changing factors, and that it will still perform, well deeper in the tree.

3 Previous Work

The dominant search techniques for MPGP can be broadly divided into stochastic methods, such as the popular UCT algorithm [12], and deterministic meth-

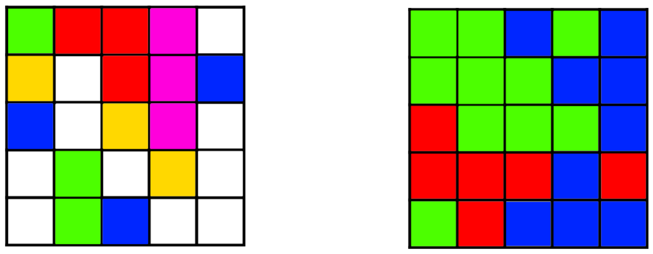

Fig. 1. The Virus Game at its initial state, and ten turns into the game. Observe that two players have been eliminated, and the pieces are more closely grouped together.

ods, generally derived from the well-known Mini-Max approach for two-player games [1, 2]. In the case of a multi-player environment, there does not exist the simple paradigm of one player maximizing his score, while the others attempt to minimize it, as one player's gain does not necessarily equally impact all opponents. This must be addressed when extending Mini-Max to a multi-player environment. The Paranoid algorithm seeks to accomplish this by considering all opponents to minimize the perspective player's score, whereas the Max-N algorithm assumes each opponent will seek to minimize his own score [5, 6]. The more recent BRS attempts to overcome the weaknesses of assuming all opponents are a coalition, as in the Paranoid approach, and the tree pruning difficulties that the Max-N algorithm encounters [8]. It does this by assuming all opponents are a single entity, who may take only one action between the perspective player's turns. While this implies it considers invalid board states, it was found to perform better than the Paranoid and Max-N in a variety of environments [8].

As the BRS groups opponents together into a single minimizing "super opponent", the Min nodes will naturally have a larger branching factor when searching the game tree than any single player, as all possible moves for all opponents are grouped together. As this grouping is a unique feature of the BRS, it was recognized that using some form of ranking of the opponents could potentially lead to a novel move ordering strategy, thus implying that one could apply methods previously unused in the context of move ordering. Using this, the present authors proposed, in [9], the Threat-ADS heuristic, which gathers moves from each opponent in the order of their relative threats. The Threat-ADS heuristic employs a list-based ADS, which is a list designed to quickly and efficiently update its structure to improve query efficiency over time [13, 14]. In this context, the list is "queried" when an opponent provides the most minimizing move at a Min node of the tree (which is discovered naturally in the execution of the BRS), and the resultant structure is interpreted as a ranking of opponent threats. Figure 2 shows a comparison of a Min level of the BRS using the Threat-ADS, and not, showing how the ADS can assist in finding the most minimizing move quickly.

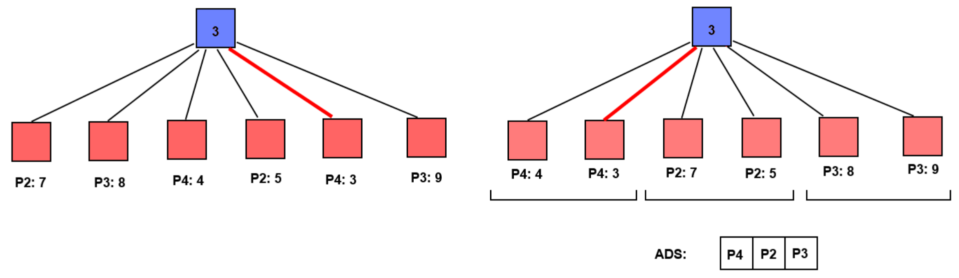

Fig. 2. The BRS not using Threat-ADS on the left, and using it, on the right. Note how the ranking of opponents by their threat levels improves move ordering.

In its introductory paper, the Threat-ADS was shown to provide statistically significant gains in terms of Node Count (NC), using a standard T-Test, over several turns of a variety of game models, using a "Move-to-Front" update mechanism for the ADS [9]. It was later shown to retain these improvements under a variety of alternative update mechanisms, although the "best" one to use can vary depending on the game in question, and to retain its performance in trees of deeper ply depth [10].

Our previous work demonstrated the benefits of Threat-ADS under a variety of game models, and using different ADS update mechanisms. However, all measurements were taken beginning at the initial board state of the game, and a few turns into the future. The question now is whether the Threat-ADS can also serve as a feasible solution to the intermediate board positions, and we shall demonstrate that it is.

4 Game Models and Experimental Setup

For the purposes of consistency with our previous work, we have elected to use the same game models that we made use of in [9] and [10]. Specifically, we make use of the well-known multi-player games Focus and Chinese Checkers, as well as a game akin to the well-known "Game of Life" called the Virus Game. Focus and Chinese Checkers were chosen due to their use in the work that introduced the BRS, to demonstrate its efficacy under a variety of game models [8]. The rules and evaluation functions for all the three games employed are the same as in the previous work in [9] and [10].

The Virus Game is a territory control game played on an N-by-N sized board, where any number of k players, where $k \geq 2$, have a configured number of pieces placed randomly on the board. During their turn, a player may "infect" a square occupied by or adjacent to (interpreted as horizontally or vertically adjacent, but not diagonally) one of their pieces. The player claims the infected square, and gains pieces on that square, and all adjacent squares. Further details of the games are omitted here in the interest of space, but can be found in [9].

Determining Intermediate Starting Configurations: In our previous work, other than the perspective player, all opponents made random moves, to cut down on experiment runtime, as we are interested in tree pruning rather than the final state of the game. However, this was not considered to be a valid way to generate intermediate starting board positions, as the intermediate positions would be unrealistic if one player was acting intelligently, and the opponents were acting at random. We thus progressed the game to an intermediate position by having each player use a simple 2-ply BRS for a set number of turns, and then switching to a 4-ply BRS for the perspective player, and random behaviour for the opponents (as in the previous work), while measurements were taken. The number of turns we advanced into the game in this way was fifteen for the Virus Game, ten for Chinese Checkers, and, given its short duration, three for Focus.

The games were run for the same number of turns as in [9] and [10], and the sum of the Node Count (NC) over those turns was taken as our metric. This was ten turns for the Virus Game, five for Chinese Checkers, and three for Focus.

In the case of the Virus Game, we set the number of players to be equal to five, and four players for Focus. In the case of Chinese Checkers, unlike in previous work, we present our results for both the four-player and the six-player case. Each of these trials was repeated fifty times, for each of the adaptive list mechanisms from [10], with the exception of the Stochastic Move-to-Rear absorbing rule, which was found to not produce any effect, and for the unaugmented BRS. These mechanisms were, specifically, the Move-to-Front, Transposition, Move-Ahead-k, and POS(k) rules (exact specification of the implementation of these update mechanisms is omitted from this work, and the reader is referred to [10]). As in previous work, the value for k was chosen to be two, to insure Move-Ahead-k and POS(k) did not perform identically to the Move-to-Front or Transposition.

In our previous work, we performed a total of two hundred trials. This was subsequently found to be very time consuming, and more than necessary to demonstrate a clear improvement from the use of the Threat-ADS heuristic. Because of the large sample, the results in [9] and [10] pass the test for normalcy, although they do not pass it in this work. Thus, rather than make use of the one-tailed T-Test, we have instead elected to employ the non-parametric Mann-Whitney test for statistical significance. We have furthermore included the Effect Size measure, which is the standardized mean difference between the two data sets, to make the degree of impact from the Threat-ADS more intuitive [15].

Our results and statistical analysis are presented in the next section.

5 Results

Consider Table 1, where we present our results for the Virus Game. We observe that, as was the case in [10], all ergodic ADSs produced a statistically significant improvement in pruning, ranging from a 5% – 10% reduction in NC. The Move-to-Front and Transposition performed the best, and equally well, in this case. The Effect Sizes ranged between approximately 0.5 and 0.9, with the majority near 0.8.

Table 1. Results for the Virus Game

Update Mechanism	Avg. NC	Std. Dev	P-Value	Effect Size
None	303,000	$35,000$	-	-
Move-to-Front	275,000	$40,000$	2.5×10^{-4}	0.82
Transposition	275,000	$37,000$	6.0×10^{-5}	0.87
Move-Ahead-k	280,000	$30,000$	1.4×10^{-4}	0.72
POS(k)	286,000	$40,000$	5.7×10^{-3}	0.56

Table 2 shows our results for Focus. In this case, again, all adaptive update mechanisms produced a statistically significant improvement in tree pruning. In this case it was a very large reduction of roughly 20% for all cases. However, here the Effect Size lingered around 0.45, given the much larger variance of the datasets. As there are a total of three opponents, the Move-Ahead-k and POS(k) rules are identical to the Move-to-Front and Transposition strategies, respectively, and thus the results are duplicated in the table, and marked '*'.

Table 2. Results for Focus

Update Mechanism	Avg. NC	Std. Dev	P-Value	Effect Size
None	16,767,000	$7,161,000$	-	-
Move-to-Front	13,592,000	$5,240,000$	0.01	0.44
Transposition	13,671,000	$4,240,000$	0.03	0.43
Move-Ahead-k*	13,592,000*	$5,240,000$	0.01	0.44
POS(k)*	13,671,000*	$4,240,000$	0.03	0.43

Our results for four-player Chinese Checkers are shown in Table 3. Again, the Move-to-Front and Transposition are identical to Move-Ahead-k and POS(k), respectively. Here we saw a statistically significant improvement from the Move-to-Front/Move-Ahead-k rule (a 13% reduction in tree size), however, while the average NC was reduced, the Transposition/POS(k) fell outside 95% certainty.

Table 4 shows our results for six-player Chinese Checkers. In this case, Threat-ADS had a smaller impact than the others. However, we do observe that the average NC is lower when Threat-ADS is employed in all cases. The best performance was obtained from the Transposition rule, with an 8% reduction in tree size, which fell within 90% certainty, although outside 95% certainty.

Table 3. Results for four-player Chinese Checkers

Update Mechanism	Avg. NC	Std. Dev	P-Value	Effect Size
None	3,458,000	$905,000$	-	-
Move-to-Front	3,024,000	$816,000$	0.01	0.46
Transposition	3,206,000	$798,000$	0.12	0.26
Move-Ahead-k*	3,024,000*	$816,000$	0.01	0.46
POS(k)*	3,206,000*	$798,000$	0.12	0.26

Table 4. Results for six-player Chinese Checkers

Update Mechanism	Avg. NC	Std. Dev	P-Value	Effect Size
None	8,168,000	$2,560,000$	-	-
Move-to-Front	7,677,000	$2,280,000$	0.18	0.30
Transposition	7,494,000	$1,670,000$	0.09	0.26
Move-Ahead-k	7,644,000	$1,830,000$	0.21	0.20
POS(k)	7,975,000	$2,190,000$	0.44	0.08

6 Discussion

Our results in this work further reinforce the conclusions from [9] and elaborated upon in [10]. In the case of each of our game models, we see that the use of Threat-ADS resulted in a decrease in the average NC, over the course of several turns. In the case of the Virus Game and Focus, the results were always statistically significant within a 95%, or greater, certainty threshold. While we always see an improvement in the case of Chinese Checkers, in a number of cases this improvement falls outside 95% certainty. *These results demonstrate that the Threat-ADS is able to produce meaningful savings in tree pruning in a variety of intermediate game states, for the games we have analyzed.* Given that the games vary substantially from each other, being based on piece capturing, racing, and territory control for Focus, Chinese Checkers, and the Virus Game, respectively, we hypothesize that the Threat-ADS will continue to perform well in other multi-player games too.

We observe that in our previous experiments, the improvement in tree pruning would range between a 5% and 10% reduction in average NC [10]. Our results for the Virus Game in this work is consistent with that. However, we note that in the context of Focus, we observed a much stronger 20% reduction in average NC. We suspect the reason for this is that, after some time is passed, it is likely that one or more of the opponents will have performed well in the opening moves of the game, and thus pose a greater threat to the perspective player. By prioritizing that opponent, the Threat-ADS is therefore able to make substantial savings, whereas at the beginning of the game, all players start at equal footing. This demonstrates that in fact the benefits of the Threat-ADS can improve over the course of the game, generating greater savings as time progresses.

In the case of Chinese Checkers, our best results for four-player Chinese Checkers resulted in a 13% saving in tree pruning, and the Transposition rule generated an 8% reduction in NC in the six-player case, which is a greater reduction than was observed in [10]. However, the best result for six-player Chinese Checkers fell outside 95% statistical certainty, regardless of the greater reduction in tree size. We suspect this is due to the large variance, which is to be expected, given that we are beginning measurements from a different starting position each time. Given that the use of Threat-ADS produces a reduction in tree size at all times, we suspect that it is not the case that the Threat-ADS is performing worse in the context of intermediate board states, but simply that the increased variance makes each pairwise comparison less likely to be statistically significant within 95% certainty. This also leads to a reduced Effect Size metric.

The Effect Size provides an easily-understood metric for demonstrating the degree of impact that a new technique has, in a way that is immediately obvious to the reader. While these rules are not universal, an Effect Size of 0.2 is considered to be small, 0.5 to be medium, and 0.8 to be large [15]. Given that our best-performing strategies are normally between 0.5 and 0.8 or larger, we conclude that Threat-ADS' contribution to tree pruning is not coincidental.

7 Conclusions and Future Work

In this paper, we have considered the task of designing an efficient AI scheme for multi-player games analyzed from intermediate (as opposed to starting) board states. Our results clearly demonstrate that the scheme presented in [9] and [10], i.e., the Threat-ADS, is an expedient strategy. Indeed, it maintains its performance when applied to intermediate board states, and does not function disproportionately well at the start of the game. In fact, our results confirm that the Threat-ADS can obtain greater performance in some cases later in the game, as we observed with Focus.

The Virus Game, Focus, and Chinese Checkers are different enough from each other that one can reasonably hypothesize that the Threat-ADS will remain viable under other game models. However, we have currently not explored its function in a very wide set of games, and thus exploration of a larger set of games is a potential future research direction. We have also shown here that Threat-ADS' performance within Chinese Checkers varies depending on whether the game is played with four or six players, showing that examination of different numbers of players may lead to interesting results as well.

As in our previous work, however, what the Threat-ADS' consistent results, with its inexpensive cost, demonstrates is the potential benefits ADSs hold for game playing. Hopefully, this work will inspire others and provide a basis for further exploration of ADS-based techniques within area of game playing.

References

1. S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, pp. 161–201. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 3rd ed., 2009.
2. N. Sturtevant, *Multi-Player Games: Algorithms and Approaches.* PhD thesis, University of California, 2003.
3. I. Szita, C. Guillame, and P. Spronck, “Monte-carlo tree search in settlers of catan,” in *Proceedings of ACG’09, the 2009 Conference on Advances in Computer Games*, pp. 21–32, 2009.
4. N. Sturtevant and M. Bowling, “Robust game play against unknown opponents,” in *Proceedings of AAMAS’06, the 2006 International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 713–719, 2006.
5. C. Luckhardt and K. Irani, “An algorithmic solution of n-person games,” in *Proceedings of the AAAI’86*, pp. 158–162, 1986.
6. N. Sturtevant, “A comparison of algorithms for multi-player games,” in *Proceedings of the Third International Conference on Computers and Games*, pp. 108–122, 2002.
7. N. Sturtevant, M. Zinkevich, and M. Bowling, “Prob-Maxn: Playing n-player games with opponent models,” in *Proceedings of AAAI’06, the 2006 National Conference on Artificial Intelligence*, pp. 1057–1063, 2006.
8. M. P. D. Schadd and M. H. M. Winands, “Best Reply Search for multiplayer games,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, pp. 57–66, 2011.
9. S. Polk and B. J. Oommen, “On applying adaptive data structures to multi-player game playing,” in *Proceedings of AI’2013, the Thirty-Third SGAI Conference on Artificial Intelligence*, pp. 125–138, 2013.
10. S. Polk and B. J. Oommen, “On enhancing recent multi-player game playing strategies using a spectrum of adaptive data structures,” in *In Proceedings of TAAI’2013, the 2013 Conference on Technologies and Applications of Artificial Intelligence*, 2013.
11. M. Levene and J. Bar-Ilan, “Comparing typical opening move choices made by humans and chess engines,” *Computing Research Repository*, 2006.
12. S. Gelly and Y. Wang, “Exploration Exploitation in Go: UCT for Monte-Carlo Go,” in *Proceedings of NIPS’06, the 2006 Annual Conference on Neural Information Processing Systems*, 2006.
13. T. H. Corman, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, pp. 302–320. Upper Saddle River, NJ, USA: MIT Press, 3rd ed., 2009.
14. J. H. Hester and D. S. Hirschberg, “Self-organizing linear search,” *ACM Computing Surveys*, vol. 17, pp. 285–311, 1985.
15. R. Coe, “It’s the effect size, stupid: What effect size is and why it is important,” in *Annual Conference of the British Educational Research Association*, (University of Exeter, Exeter, Devon), 2002.