



Games, computers, and artificial intelligence

Jonathan Schaeffer^{a,*}, H. Jaap van den Herik^b

^a *Department of Computing Science, University of Alberta, Edmonton, AB, Canada T6G 2E8*

^b *Department of Computer Science, Universiteit Maastricht, PO Box 616,
6200 MD Maastricht, Netherlands*

1. Introduction

In 1950, Claude Shannon published his seminal paper on computer chess [23]. It was the dawn of the computer age; ENIAC was only a few years old, and visionary people like Shannon and Alan Turing could see the tremendous potential for the technology. Many computers in that era were used for military applications, typically ballistic calculations for missiles. In contrast, games seemed to be a natural application for computers, and one that an average person could relate to. A desire to use computers for applications that would attract public attention motivated Arthur Samuel to begin his 25-year quest to build a strong checkers-playing program [18,20]. The first working checkers program appeared in 1952 [27], and chess programs followed shortly thereafter [10].

The early efforts of Shannon, Samuel, Turing, Allan Newell, Herbert Simon, and others generated considerable interest in researching computer performance at games. Developing game-playing programs became a major research area in the fledgling field of artificial intelligence (AI). Indeed, building a world-championship-caliber chess program was one of the original “grand challenge” applications for AI. At the time, few realized the difficulty of creating programs that exhibited human-level “intelligence”, and the early days of AI were plagued by over-optimistic predictions [25].

In the 1970s and 1980s, computer-games research concentrated on chess and the so-called brute-force approach. The success of the Northwestern University chess program (the CHESS series of programs) [26] showed there was a strong correlation between search speed and chess-program performance. This was later quantified by Ken Thompson [29]. The consequence was a prolonged period of games-development activity largely devoted to building faster search engines, at the expense of doing mainstream research. The importance of developing a high-performance chess program seemed to fade.

* Corresponding author.

E-mail addresses: jonathan@cs.ualberta.ca (J. Schaeffer), herik@cs.unimaas.nl (H.J. van den Herik).

From the late 1980s until today, computer-games research has been rejuvenated. Beginning with the publication of a series of innovative search algorithms (such as conspiracy numbers [14]), search enhancements (for example, null moves [2] and singular extensions [1]), and learning ideas (such as temporal difference learning [28]), games-related research has branched off into many exciting areas. Some of this research is captured in this volume.

After many decades of slow progress, the 1990s witnessed a series of dramatic computer successes against the best of humanity. The first game to have a non-human World Champion was checkers (8×8 draughts). In 1994, the program CHINOOK won the World Man–Machine Championship [21,22]. This was quickly followed in 1997 by DEEP BLUE’s victory over World Chess Champion Garry Kasparov [7] and LOGISTELLO’s dominating win over World Othello Champion Takeshi Murakami [3]. With many significant milestones being reached by game-playing programs, now is a good time to take stock of what has been achieved and what remains to be done. This volume presents papers on the progress of building world-championship programs for popular games and puzzles. Most of the authors are the developers of the best program in the world for that game. As the results documented in this volume show, high-performance game-playing programs have been one of AI’s most visible success stories.

The human side of building these programs is just as interesting as the technical side. The inside stories of building world-championship-caliber game-playing programs have been described for CHINOOK (checkers) [21] and DEEP BLUE (chess) [7].

2. Puzzles

This volume includes two papers illustrating different facets of building high-performance programs for solving puzzles. The 24-puzzle and crossword puzzles are the application domains used.

Richard Korf is one of the leading names in search-related research. In 1985, he introduced the IDA* algorithm, using the 15-puzzle to illustrate his ideas (search-space size is $O(10^{13})$) [11]. Solving the 15-puzzle became a routine computational exercise, one that was repeated in many AI courses. Korf and Ariel Felner’s paper “Disjoint Pattern Database Heuristics” uses new heuristics to push the sliding-tile frontier forward. Solving instances of the 24-puzzle (search-space size is $O(10^{25})$) is now going to become routine. In effect, in 16 years (from 1985 to 2001) Korf and his colleagues have improved our sliding-tile puzzle solving capabilities by an incredible factor of 10^{12} !

Michael Littman, Greg Keim, and Noam Shazeer write about “A Probabilistic Approach to Solving Crossword Puzzles”. Littman used his graduate course to tackle the problem of solving crossword puzzles, with the students each working on different aspects of the program. The result is a remarkable piece of work, combining single-agent search, probability analysis, database queries, and web searches. The amazing aspect of the work is that the program performs very well on difficult crossword puzzles *without understanding the semantics of the puzzles’ clues!*

Other interesting puzzle research includes Rubik’s Cube [12] and Sokoban [8,9].

3. Two-player perfect-information games

Five papers describe the stunning success that computers have had in two-player perfect-information games. The featured games are chess, Othello, hex, shogi and go.

The most visible success for computer game-playing occurred in May 1997 when the chess machine DEEP BLUE defeated World Chess Champion Garry Kasparov in an exhibition match. This was a major success for artificial intelligence and a milestone in computing history. Murray Campbell, Joseph Hoane Jr., and Feng-hsiung Hsu describe the architecture and implementation of their chess machine in the paper “Deep Blue”. For Campbell and Hsu, this represented the culmination of over a decade of research, development, and machine building [7].

A few months after the chess success, Othello became the third game to fall to computers when Michael Buro’s program LOGISTELLO defeated the World Othello Champion Takeshi Murakami. LOGISTELLO repeatedly played itself to learn over one million evaluation function weights. In the paper “Improving Heuristic Mini-Max Search by Supervised Learning”, Buro discusses the learning algorithms used in his program.

Hex is a relatively new game (invented in the 1940s), but its simplicity of rules gives rise to elegant and intricate play. In “A Hierarchical Approach to Computer Hex”, Vadim Anshelevich describes his program HEXY. The program uses a novel way of searching the tree, using a theorem-prover-like search to evaluate leaf nodes. HEXY is competitive with top human players, but mankind still holds a significant edge (especially on larger board sizes).

Shogi is often referred to as Japanese chess and, indeed, it shares many similarities with its Western counterpart. However, the ability to return captured pieces to the board results in a more tactical game with an increased branching factor. Hiroyuki Iida, Makoto Sakuta, and Jeff Rollason describe the state of the art for this game in their paper “Computer Shogi”. Programs have a long way to go before they can offer a serious challenge to a shogi grandmaster. The authors predict this will happen by the year 2010.

For many years, chess was considered the *drosophila* (fruit fly) of artificial intelligence (chess is to AI as the fruit fly is to genetics). With the chess “problem” now out of the way, the Oriental game of go has emerged as a major challenge to artificial-intelligence research. Martin Müller describes the current state and future challenges in his paper “Computer Go”. The best programs are still weak players, and this situation is likely to remain so for quite a while, despite increased research efforts.

Other two-person perfect-information game milestones include strong programs for:

- (1) Draughts (10 × 10 checkers): Draughts programs are strong, but not yet a serious threat to the World Champion. However, recently (August 2001) the human Grandmaster Samb faced considerable difficulties in his match against the program BUGGY.
- (2) Lines of Action: This is a relatively new game where the object is for each side to connect all their pieces together. The program MONA recently won the Fifth Annual Lines of Action E-mail Tournament against the world’s best human players. Several programs, including MONA, YL [16], and MIA [30] are considered to be better than all human players.

- (3) Amazons: Amazons is being touted as a challenge that nicely bridges the gap between chess and go. Like go, the object is to capture territory. Programs benefit from chess-like searches without the need for extensive knowledge, but suffer from the large branching factor (the starting position has 2,176 legal moves!). The best programs are still quite weak [6].

4. Imperfect-information and stochastic games

Programming games like chess is “easy” in the sense that the program has access to the entire state of the game. In some games, information is hidden (e.g., you cannot see the opponent’s cards) and the program has to account for all possible scenarios, only one of which is actually valid. This fundamentally changes the search space, usually resulting in a large branching factor and the need to account for probabilities. Other games are stochastic, in that there is an element of chance (such as dice rolls). Again, there may be a large number of possibilities, only one of which will occur. Techniques that work for two-player perfect-information games do not carry over to games that have imperfect information or are stochastic. These ideas are illustrated using backgammon, poker, and Scrabble.

At the 1998 AAI Hall of Champions, Gerry Tesauro’s backgammon program TD-GAMMON matched wits with World Champion Malcolm Davis. The human narrowly edged the computer in the 100-game match. Since then, there is increasing evidence that the best computer programs have surpassed the best human players at backgammon. Tesauro’s use of temporal difference learning to tune his program’s evaluation weights was a milestone in the machine-learning community. In “Programming Backgammon Using Self-Teaching Neural Nets”, Tesauro describes some of the innovative aspects of his program.

Building programs to play poker well is very challenging from the AI point of view. Programs must deal with incomplete knowledge (hidden opponent cards), multiple competing agents (10 players per game), risk management (betting), opponent modeling (exploiting sub-optimal opponent play), and deception (bluffing). In “The Challenge of Poker”, Darse Billings, Aaron Davidson, Jonathan Schaeffer, and Duane Szafron describe their program POKI. The program plays well, but a considerable performance gap remains to be overcome. The dream of winning in Las Vegas is still a few years away.

Brian Sheppard gives the first in-depth description of a world-class Scrabble program in his paper “World-Championship-Caliber Scrabble”. Sheppard’s program MAVEN has had impressive results against the best human players. He writes that [24]:

“The evidence right now is that MAVEN is far stronger than human players. . . . I have outright claimed in communication with the cream of humanity that MAVEN should be moved from the ‘championship caliber’ class to the ‘abandon hope’ class, and challenged anyone who disagrees with me to come out and play. No takers so far, but maybe one brave human will yet venture forth.”

No one has.

One popular game that is not addressed in this volume is bridge. Matt Ginsberg’s GIB program is a strong master, and has acquitted itself well against human World Champions.

Within five years a world-championship-caliber bridge program will likely exist. Excellent descriptions of GIB can be found in [4,5].

5. Perfect programs

The board and card games described in this volume are fun because of the intellectual challenges that they offer. Humans love to unravel the mysteries of games, seeking beauty and mathematical truth. In some cases it is possible to develop a computer program that can play perfectly; it is an oracle that can reveal all the secrets of the game. The last paper deals with the ultimate challenge of a puzzle or game: solving it. In “Games Solved: Now and in the Future”, Jaap van den Herik, Jos Uiterwijk, and Jack van Rijswijk discuss programs that can play games or subsets of games perfectly. The authors highlight the techniques used so far, and offer predictions for which games will be solved in the near future.

6. New frontiers

In the realm of board and card games, go will continue to taunt AI researchers for many decades to come. As well, new games will come along to provide interesting challenges. For example, the game of Octi was invented to be resistant to computer algorithms [15]. It is characterized by having a large branching factor, making deep search impractical. However Octi has the additional dimension that a move can change the capabilities of a piece, making it challenging to design an evaluation function.

The research into board and card games is, in some sense, historically motivated because these were posed as interesting challenges in the early years of computing. However, with the advent of home computers, new forms of computer games have emerged, supported by a \$10 billion (and growing) industry: interactive computer games. There are numerous products on the market covering the gamut of action games (e.g., shoot'em-up games), role-playing games (e.g., player goes on a quest), adventure games (e.g., navigating through a scripted story), strategy games (e.g., controlling armies in a war), “God” games (e.g., evolving a simulated population), and sports (e.g., controlling a player or coaching a team) [13]. Historically, these games have been long on graphics, and short on artificial intelligence.¹

John Laird promotes interactive computer games as an opportunity for the AI research community [13]. Many interactive computer games require computer characters that need to interact with the user in a realistic, believable manner. Computer games are the ideal application for developing human-level AI. There is already a need for it, since human game players are generally dissatisfied with computer characters. The characters are shallow, too easy to predict, and, all too often, exhibit artificial stupidity rather than artificial intelligence. This has led to the success of on-line games, where players compete against other humans. The current state of the art in developing realistic characters can

¹ For example, path finding is a critical component of many games, yet it took until 1996 for the industry to “discover” A*.

be described as being primitive, with simple rule-based systems and finite-state machines being the norm. The lack of sophistication is due to the lack of research effort (and, cause and effect, research dollars). This is changing, as more games companies and researchers recognize that AI will play an increasingly important role in game design and development. The quality of the computer graphics may draw you to a product, but the play of the game will keep you using the product (and buying the sequel). Artificial intelligence is critical to creating a satisfying gaming experience.

Finally, the last few years have seen research on team games become popular. The annual RoboCup competition encourages hardware builders and software designers to test their skills on the soccer field [17].

7. Computers and games

The work on computer games has resulted in advances in numerous areas of computing. One could argue that the series of computer-chess tournaments that began in 1970 and continue to this day represents the longest running experiment in computing-science history. Research using games has demonstrated the benefits of brute-force search, something that has become a widely accepted tool for a number of search-based applications. Many of the ideas that saw the light of day in game-tree search have been applied to other algorithms. Building world-championship-caliber games programs has demonstrated the cost of constructing high-performance artificial-intelligence systems. Games have been used as experimental test beds for many areas of artificial intelligence. And so on.

Arthur Samuel's concluding remarks from his 1960 paper are as relevant today as they were when he wrote the paper [19]:

“Programming computers to play games is but one stage in the development of an understanding of the methods which must be employed for the machine simulation of intellectual behavior. As we progress in this understanding it seems reasonable to assume that these newer techniques will be applied to real-life situations with increasing frequency, and the effort devoted to games . . . will decrease. Perhaps we have not yet reached this turning point, and we may still have much to learn from the study of games.”

Acknowledgements

Jonathan Schaeffer's research was supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) and Alberta's Informatics Circle of Research Excellence (iCORE). Jaap van den Herik acknowledges the support of the Netherlands Organization for Scientific Research (NWO) and the Foundation for Scientific Education in Limburg (SWOL). The editors want to extend their heart-felt thanks to the paper authors and the anonymous referees for helping make this an outstanding volume. None of this would have been possible without the inspiration and support from Elsevier. We are grateful to Arjen Sevenster and Zeger Karssen.

References

- [1] T. Anantharaman, M. Campbell, Singular extensions: Adding selectivity to brute-force searching, *Artificial Intelligence* 43 (1990) 99–109.
- [2] D. Beal, A generalised quiescence search algorithm, *Artificial Intelligence* 43 (1990) 85–98.
- [3] M. Buro, The Othello match of the year: Takeshi Murakami vs. Logistello, J. Internat. Computer Chess Assoc. 20 (3) (1997) 189–193.
- [4] M. Ginsberg, GIB: Steps toward an expert-level bridge-playing program, in: T. Dean (Ed.), Proc. IJCAI-99, Stockholm, Sweden, Vol. 1, Morgan Kaufmann, San Francisco, CA, 1999, pp. 584–589.
- [5] M. Ginsberg, GIB: Imperfect information in a computationally challenging game, *J. Artificial Intelligence Res.* 14 (2001) 303–358.
- [6] T. Hashimoto, Y. Kajihara, N. Sasaki, H. Iida, J. Yoshimura, An evaluation function for amazons, in: H.J. van den Herik, B. Monien (Eds.), *Advances in Computer Games*, Vol. 9, Universiteit Maastricht, Maastricht, 2001, pp. 191–203.
- [7] F.-h. Hsu, *Behind Deep Blue*, Princeton University Press, Princeton, NJ, 2002.
- [8] A. Junghanns, New developments in single-agent search, Ph.D. Thesis, Department of Computing Science, University of Alberta, Edmonton, AB, 1999.
- [9] A. Junghanns, J. Schaeffer, Enhancing single-agent search using domain knowledge, *Artificial Intelligence* 129 (1–2) (2001) 219–251.
- [10] J. Kister, P. Stein, S. Ulam, W. Walden, M. Wells, Experiments in chess, *J. ACM* 4 (1957) 174–177.
- [11] R. Korf, Depth-first iterative-deepening: An optimal admissible tree search, *Artificial Intelligence* 27 (1985) 97–109.
- [12] R. Korf, Finding optimal solutions to Rubik’s Cube using pattern databases, in: Proc. AAAI-97, Providence, RI, 1997, pp. 700–705.
- [13] J. Laird, M. van Lent, Human-level AI’s killer application: Interactive computer games, in: Proc. AAAI-2000, Austin, TX, 2000, pp. 1171–1178.
- [14] D. McAllester, Conspiracy numbers for min-max search, *Artificial Intelligence* 35 (1988) 287–310.
- [15] Octi, www.octi.net.
- [16] University of Alberta GAMES Group, www.cs.ualberta.ca/~games.
- [17] RoboCup, www.roboocup.com.
- [18] A. Samuel, Some studies in machine learning using the game of checkers, *IBM J. Res. Develop.* 3 (1959) 210–229.
- [19] A. Samuel, Programming computers to play games, in: F. Alt (Ed.), *Advances in Computers*, Vol. 1, Academic Press, New York, 1960, pp. 165–192.
- [20] A. Samuel, Some studies in machine learning using the game of checkers: Recent progress, *IBM J. Res. Develop.* 11 (1967) 601–617.
- [21] J. Schaeffer, *One Jump Ahead: Challenging Human Supremacy in Checkers*, Springer, Berlin, 1997.
- [22] J. Schaeffer, J. Culberson, N. Treloar, B. Knight, P. Lu, D. Szafron, A world championship caliber checkers program, *Artificial Intelligence* 53 (2–3) (1992) 273–289.
- [23] C. Shannon, Programming a computer for playing chess, *Philosophical Magazine* 41 (1950) 256–275.
- [24] B. Sheppard, Personal communication to J. Schaeffer.
- [25] H. Simon, A. Newell, Heuristic problem solving: The next advance in operations research, *Oper. Res.* 6 (1958) 10.
- [26] D. Slate, L. Atkin, Chess 4.5—The Northwestern University chess program, in: P. Frey (Ed.), *Chess Skill in Man and Machine*, Springer, Berlin, 1977, pp. 82–118.
- [27] C. Strachey, Logical or non-mathematical programmes, in: Proc. Association for Computing Machinery Meeting, Toronto, ON, 1952, pp. 46–49.
- [28] G. Tesauro, Practical issues in temporal difference learning, *Machine Learning* 8 (1992) 257–277.
- [29] K. Thompson, Computer chess strength, in: M. Clarke (Ed.), *Advances in Computer Chess* 3, Pergamon Press, Oxford, 1982, pp. 55–56.
- [30] M. Winands, J. Uiterwijk, J. van den Herik, The quad heuristic in Lines of Action, *J. Internat. Computer Chess Assoc.* 24 (1) (2001) 3–15.