The World Championship Checkers (WCC) Story

by Gil Dodgen — July 1997

Checkers, known in the British Isles as draughts (pronounced “drafts”), is not only one of the oldest games known (checker pieces and boards have been found in Egyptian tombs), but a little experience and study will reveal that, despite its apparent simplicity, it is a game of tremendous beauty, elegance, depth and complexity.

***

Almost everyone is aware that a tremendous amount of work, by numerous brilliant people over several decades, has gone into computer chess, and that master-level chess machines and programs have been around for years. However, few are aware that some of the earliest artificial intelligence research involved the game of checkers, and that master-level checker programs are a recent achievement.

In the late 1950’s and early 1960’s Arthur Samuel did pioneering work in artificial intelligence using the game of checkers. His interest was in machine learning, which did not pan out too well.

Despite claims made about his program, it was a weak amateur. A look at the games it played reveals that it was decent at tactics, but clearly lacked knowledge of sound positional play. Unfortunately, his program did beat an “expert” player in one game, and so began the rumor that the game had been mastered by computers. This “expert” was a blind man, not a particularly great player, and the loss resulted from a simple oversight on his part.

It was this discovery about Samuel’s program that piqued my interest.

I earn my living as editor of Hang Gliding magazine. (I took up the sport while in college.) In the late 1980’s I got my first computer, an Amiga, to do word processing. (Believe it or not, up until that time all editing was done with a pencil and a typewriter, after which the hard copy was given to a typesetter for keyboarding.) Strangely enough, my college degrees are in music and foreign language (piano and French) and I had never taken a computer class, but the Amiga came with a Basic interpreter and I soon became enthralled with programming.

After a few months of writing simple game-playing programs I became intrigued with the idea of artificial intelligence (AI) and picked up some books on the subject. It quickly became obvious that far too much work had been done on chess for me to ever make a significant contribution, but the books mentioned Samuel’s early checkers AI project, and, as far as I could tell, no further work had been done on the game in the intervening quarter century.

So, I set about implementing the basics on the Amiga: a move generator, an Alpha-Beta tree search, and an evaluator. The program was doing three- or four-ply searches, and I was thrilled one day when it caught me in a shot and won. At this point I had exhausted what I had learned from the books, and not being a decent checker player or a trained computer programmer, I knew that I would need some help.

One day I found yet another AI book that mentioned a checker program written in 1979 by two graduate students, Tom Truscott and Eric Jensen, at Duke University. I called the Duke computer science department and tracked down Tom, who had long since left the campus. He sent me technical papers on their project and gave me many ideas for improving the performance of the program. By this point in time, teams designing chess programs had published more papers on their incredible advances in brute-force tree searching techniques, and most of these applied nicely to checkers.

Now I realized that Amiga Basic would no longer be suitable for this high-powered programming, so I picked up The Waite Group's C Primer and took it with me on a vacation that involved many hours in a car. I devoured the book when not driving, and came home with a new computer language under my belt. In 1988 desktop publishing took off and I bought a Macintosh II for this purpose. I also purchased a C compiler and ported the program to the Mac.

The big problem at this point was that I had no idea how good the program really was, since it was beating me consistently, so I tracked down the American Checker Federation (ACF). I was thrilled to discover that two of the nation’s top Master players lived nearby (in Southern California terms). Ed Markusic and Leo Levitt soon became my checker mentors, helping me refine the program’s evaluation function.

I vividly remember my first meeting with Ed. Based on Checkers 1.0’s play against me I was convinced that I had a powerhouse on my hands, and I knew my program was vastly out-searching Samuel’s. (Keep in mind the primitive hardware and searching techniques Samuel used in the 1950's and early 1960's. My Mac II had roughly the computing power of the $3,000,000 IBM 370/165 on which the 1979 Duke program ran.) Alas, Ed had no trouble trouncing Checkers 1.0, and would make comments like, “See that hole there? It will never recover from that,” or, “It let me weaken its double corner; you can't do that.” Clearly more was needed than brute force.

Ed and Leo would come by my office and play the program often, pointing out strategic and positional problems. I would make mental notes and later go to work refining the evaluator until Checkers understood and played the test positions properly. (The “evaluator” is the portion of the program that assigns a score to positions at the end of the search, which must inevitably be stopped because the number of possible lines of play grows exponentially with increased search depth.)

In the spring of 1989 I got a call from Tom Truscott. David Levy of chess and computer chess fame was sponsoring the first Computer Olympiad in London (a tournament for intelligent computer programs), and Levy

wondered if Tom might be able to resurrect the Duke program to enter in the competition. Unfortunately, this was not practical since the Duke software was written in machine language for a now-ancient IBM mainframe. However, Tom thought of me and suggested that I enter Checkers 1.0.

It turned out that there was another software developer in New York, Dave Butler, who had been working on a checkers program. Dave is an expert player as well as a programmer, and we became good friends. We traded many ideas and ended up entering our brainchildren in the Olympiad.

At about this time Jonathan Schaeffer of the University of Alberta at Edmonton discovered what I had about computer checkers: it had been left in its infancy in the 1960's. Jonathan was one of the world's leading chess programmers, widely known for his powerful Phoenix program. However, dedicated chess hardware had made it impossible for him to compete any longer in the computer chess arena, so he turned his attention to checkers. He is a good friend of David Levy's, and showed up at the Olympiad with baby Chinook, a hurriedly developed checker program that ran on a Sun SPARC machine. My program ran on a Mac II and Dave Butler's ran on an old eight-bit Atari. Altogether there were six programs entered in the checkers (draughts) division. Chinook took the gold medal, Checkers 1.0 the silver, and Dave's program the bronze. At David Levy's request I warmed up my fingers for the piano and had the honor of playing the Chopin A-flat Polonaise and Fantasy Impromptu at the closing ceremonies.

Being a researcher, Jonathan was not interested in producing a commercial product (neither was I at that point) and he was more than happy to share many of the sophisticated techniques used in Chinook. Computer chess researchers had made even more stunning advances in tree searching and endgame databases since the Duke days. Jonathan sent me many technical papers and offered much help over the phone and in person.

In the spring of 1990 the University of Alberta hosted a checkers AI conference, and Leo Levitt and I attended along with several others representing the American Checker Federation. Plans were made to enter Chinook and my second-generation program, Checkers Experimental, in the 1990 U.S. National Checker Championships to be held that summer.

I was fortunate to be sponsored at the Nationals by Mips Computer Systems with an M-120, a 10,000,000-instruction-per-second Unix machine. There are too many stories to tell about the 1990 Nationals, but you can read about much of what went on in Jonathan Schaeffer's book, One Jump Ahead — Challenging Human Supremacy in Checkers, published by Springer-Verlag. We entered our programs in the Masters division (there are also Majors and Minors divisions), and in the end Chinook placed second and Checkers Experimental tied 6th/7th in a field of about 30 human Masters.

During the tournament my mentor Ed Markusic was seeded against the program he helped refine, and at one point in the four-game round he commented, "This thing is getting good!" Three draws were played, but in the final game Checkers Experimental and the screaming Mips machine got the edge and beat Ed. The Master had been defeated by his silicon student.

In the final round of the tournament Checkers Experimental went up against Marion Tinsley, universally recognized as the greatest player who ever lived. You can read more about the amazing Marion in Jonathan's book. The program played three drawn games but lost the fourth in the endgame. Endgames in both chess and checkers are tough for computer programs, because long-range strategy becomes more important than tactics. It was time to start computing endgame databases.

Over the next couple of years I continued to refine the program and compute endgame databases. The databases are generated essentially by playing the game backwards, starting with two pieces, and saving the results as won, lost or drawn for every possible position and piece combination. As you can imagine, the number of positions and the computation time grow astronomically as pieces are added. There are more than seven million possible checker positions with only four pieces on the board, and more than 2.5 billion with six pieces! In computing my databases I decided to go one step further than the Chinook team had. Their databases only contain scores of won, lost or drawn for the positions. No information as to what move to make is saved; the program must find the right move by searching as it normally does after reaching a position it previously knew was a win, loss or draw. I computed "perfect play" for the four-piece and fewer databases, but quit at this point since the complexity and time required made it impractical to go any further. The Chinook team eventually computed the eight-piece database (more than 406 billion positions) which took years - thousands of man hours and hundreds of thousands of hours of computer time.

In 1992 Checkers 3.0 was ready for the Nationals (they are held every two years), this time running on a 486/33 with its new databases. Chinook entered again, this time running over the phone lines to Canada on an eight-processor Silicon Graphics machine with much of the eight-piece database computed.

In the first round Ed Markusic stunned everyone by beating Chinook! He had purchased a 486 and had been playing my Checkers 3.0 program for some time. Ed was retired, spent a lot of time with my program, and had found some weaknesses. He later told me he had discovered that Checkers 3.0 did not know how to properly play certain bridge endings. (Ed probably kept this from me in hopes of getting revenge for the defeat in 1990. I don't blame him.) These endings can involve many pieces and are often very subtle and deep. Sure enough, Ed found an opportunity to lure Chinook into a bridge ending that he knew my program would likely misplay. Chinook had the same weakness — it was all over but the shouting. In the second round Chinook lost again! By this time most of the top players had checker programs, and were apparently getting wiser concerning the strengths and weaknesses of these beasts.

Ed was never seeded against my program, and in the end Checkers 3.0 placed third and Chinook placed sixth — my great claim to fame!

In 1993 I produced Cornell Checkers, the fourth generation of the program, which was distributed briefly by Cornell Computer Systems before

they went out of business. Cornell was basically Checkers 3.0 with a lot of new features and graphics. After the Cornell debacle I retired from checkers programming in order to concentrate on magazine publishing and raising two beautiful little daughters.

***

Jonathan Schaeffer named his chess program Phoenix because it rose from the ashes of a previous effort. Perhaps WCC should have been given the same name, because in 1996 my brainchild was to be raised from the dead.

One day in the summer of 1996 this guy named Ed Trice called me up, having tracked me down through the Chinook Web page. He understood that I had written a checkers program for the Mac, and was interested in doing the same. I told him that it was a much bigger project than he probably imagined, and sent him a copy of Cornell. Ed is a Master chess player and had written a master-level chess program in his late teens. He has also written and published an award-winning Blackjack program for the Mac. Ed quickly became enthralled with the game of checkers, and has become a master-level player in an amazingly short period of time. He suggested that we join forces and produce World Championship Checkers for both the PC and the Mac. A new project was underway.

WCC is now the fifth generation of the program. Over the last year the artificial intelligence engine has been further refined and the five- and six-piece databases added, thanks to Rob Lake and Jonathan Schaeffer of the Chinook team. New features and graphics have also be added. Its book is more than 12 times larger than Cornell’s, thanks to Al Lyman, 1996 U.S. Mail Play Champion.

In 1992 Checkers 3.0 was given an American Checker Federation rating of 2606 based on its performance in the two Nationals. We feel that WCC is now probably well into the 2700’s when running on state-of-the-art hardware under tournament time control. Marion Tinsley, the now-deceased greatest player of all time, was rated at 2812. In the near future I plan to compute perfect play for the five- and six-piece databases, the data for which will reside on a CD. This should produce some very fascinating results.

So there you have it. When you play WCC you can think about the long journey the program has taken, and the digital memories that are engraved in its little silicon soul. If WCC could talk it would have many stories to tell.