Jonathan Schaeffer

One Jump Ahead

Computer Perfection at Checkers

Revised Edition

Springer

One Jump Ahead

Computer Perfection at Checkers

Revised Edition

One Jump Ahead

Computer Perfection at Checkers

Revised Edition

Jonathan Schaeffer

Springer

Author
Jonathan Schaeffer
Department of Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2E8
jonathan@cs.ualberta.ca

ISBN: 978-0-387-76575-4 e-ISBN: 978-0-387-76576-1
DOI: 10.1007/978-0-387-76576-1

Library of Congress Control Number: 2008931408

Printed on acid-free paper

springer.com

Steph and Rebecca
for their patience and love throughout.
It’s finally over. Really!

Preface

It's hard to believe that it's been over a decade since *One Jump Ahead: Challenging Human Supremacy at Checkers* was published. I'm delighted to have the opportunity to update and expand the book. The first edition ended on a sad note and that was reflected in the writing. It is now eleven years later and the project has come to a satisfying conclusion. Since its inception, the checkers project has consumed eighteen years of my life— twenty if you count the pre-CHINOOK and post-solving work. It's hard for me to believe that I actually stuck with it for that long. My wife, Steph, would probably have something witty to say about my obsessive behavior.

Rereading the book after a decade was difficult for me. When I originally wrote *One Jump Ahead*, I vowed to be candid in my telling of the story. That meant being honest about what went right and what went wrong. I have been criticized for being hard on some of the characters. That may be so, but I hope everyone will agree that the person receiving the most criticism was, justifiably, me. I tried to be balanced in the storytelling, reflecting things as they really happened and not as some sanitized everyone-lived-happily-ever-after tale. I was appreciative of Paul Lu's comments on the book, since he apparently read my mind on this matter:

> Another strength of this book is how Jonathan fearlessly pulls no punches in presenting balanced portraits of the many people involved in the CHINOOK story. Don't expect a forgettable puff piece. Scientists and champions are not immune from human foibles. In dispelling that illusion, Jonathan tells a honest and valuable story. Nobody escapes Jonathan's (sometimes) sharp criticism: not myself (deserved), not some of the luminaries of checkers (accurate, in my opinion), and especially not Jonathan himself. But Jonathan balances this with genuine praise, affection, and respect in almost every case. Consequently, the book contains many insights on human nature, the nature of AI, and what happens when they cross paths.

With a decade between readings of the book, even I was surprised at my candor. Occasionally I had to wince and even blush at the memories that the book evoked. Did I really do that? Why on Earth would I commit that to paper and let the world know? Was I crazy? Probably.

Why should you bother reading this edition, given that you read the first edition? The new *One Jump Ahead* differs significantly:

1. Five new chapters that bring the story from 1996 to its successful conclusion in 2007. The story of solving checkers adds a new dimension to the quest to *prove* computer superiority over the incredible abilities of Dr. Marion Tinsley.
2. Seven new chapters that give the key participants a chance to tell their side of the story. The original book reflected my opinion; the other major players needed a public forum to give their recollections and express their views.
3. Additional anecdotes from 1989 to 1996 that didn't make the original cut. I don't recall why I decided to leave them out. I may have had a good reason back then, but if I can't recall it today then the reason(s) couldn't have been that important.
4. Many more photos! This helps bring some of the characters to life.
5. The text has been improved, including corrections and updates.

This book was written for a variety of audiences. First and foremost, it was intended for readers with an interest in computing science, specifically artificial intelligence and software engineering. Despite what people may think, especially when they open the cover and see diagrams of checkers positions, this book is about creating technology. It is the story of an incredibly-talented, almost-perfect checkers player, Marion Tinsley, and our attempts to create a computer program that could match his abilities. That may sound rather dry, but I like to think that the human element makes this an interesting story.

Second, I tried to write it at a level that would make it accessible to a general audience. I wanted to create a book that educated people about computing science and the challenges posed by developing complex artificial intelligence systems.

Third, the book is for checkers players. They are angry with me because I used algebraic notation for specifying checkers moves instead of their much-loved (but obscure) numeric notation. The change was an attempt to make the checkers content more accessible to a wider audience. I came to appreciate the beauty of the game, and I wanted to communicate this to as wide an audience as possible. The community is shrinking, and as a competitive sport may one day die. I hope that in my writing I have captured an accurate snapshot of the historic tradition, talented people, and collegial atmosphere that characterize this noble game.

In this book you will see the evolution of hardware technology. We go from computer speeds that went from megahertz (millions of instructions per second) to gigahertz (billions per second), computer memory that started out at megabytes (millions of bytes) and ended up at hundreds of gigabytes (hundreds of billions), and megabyte disk sizes that grew into terabytes (thousands of billions). This book is not a quaint story of old technology. The story and the lessons learned are as valuable today as they were when the events unfolded. Scientists will always push the envelope of what is technologically possible. Someone with a large problem may be starting with gigabytes (10^9) today and end up using petabytes (10^{15}) tomorrow.

I have been fortunate to be part of a unique experience. Along my eighteen year odyssey, I was privileged to work with, interact with, and compete against many wonderful people. I want to thank everyone who made this adventure possible.

June 2008 *Jonathan Schaeffer*

Acknowledgments

The CHINOOK and solving-checkers projects were team efforts from start to finish. The story is told from my point of view, but that has the disadvantage of downplaying the contributions of the other players. I have tried to rectify this, in part, by giving them a voice in this book: the key players were each asked to write a short chapter on their experiences. This adventure wouldn't have been possible without huge commitments from many people. This story, despite the many ups and downs, was a wonderful experience for me and for the members of our team. I want to extend my deepest heart-felt thanks to all of them for their selfless contributions.

The book has been enriched by the personal reflections of Martin Bryant, Neil Burch, Rob Lake, Paul Lu, Rebecca Schaeffer, Steph Schaeffer, and Norman Treloar. Thank you!

Numerous people contributed to helping make the updated version of *One Jump Ahead* possible: Darse Billings, Yngvi Björnsson, Neil Burch, Robert Holte, Andreas Junghanns, Robert Lake, Ann Nield, Rebecca Schaeffer, and Steph Schaeffer. Additional photographs were supplied (with permission) by Richard Fortman, Frederick Friedel, Richard Pask, Steph Schaeffer, Richard Siemens (University of Alberta Creative Services), and Gio Wiederhold. Rob Lake wrote the software needed to do the checkers diagrams.

I received valuable input from Richard Pask. His 1997 letter to me was full of insights and new material. His book, *The Legendary MFT*, was a valuable source of information. The book is a *tour de force* and must have been a labor of love.

Thank you to Springer-Verlag for giving me the opportunity to complete the story of my checkers saga in print. Melissa Fearon and Valerie Schofield shepherded me through the process.

A project like this would not have been possible without research funding. Alberta's Informatics Circle of Research Excellence (iCORE) and Canada's Natural Sciences and Engineering Research Council (NSERC) supported my work.

Finally, for eighteen years the Department of Computing Science at the University of Alberta provided the backdrop for this story. It has been a privilege to work in such an outstanding and collegial environment.

Preface (1997)

Why did I write this book? I'm still not sure. After all, I'm a researcher, which means I know how to write technical papers. But writing for a non-technical audience is something I know nothing about. It took a lot of effort before I could force myself to sit down to write the first word. Once I did, however, it was hard to stop! When I started this project, I didn't know that I had a lot to say and, in some sense, the results show this. The book is much longer than I ever imagined it would be. Worse yet is that there is a lot of material that I decided not to include. It's a good thing that the publishers decided to limit how long the book could be! However, after much soul searching, I think I now know why I wrote this book.

First and foremost, this book tells an interesting story. It's about the life of a checkers-playing computer program, CHINOOK, from its creation in 1989 to its retirement in 1996. In reality the story revolves around two people with different views of the program. As the creator of CHINOOK, I wanted to push the program to become the best player in the world, in much the same way that a father might encourage his son to excel at sports. The world checkers champion, Marion Tinsley, saw the program as a threat to his incredible playing record and his legitimate claim to be the best player ever. The result was a public battle between man and machine for supremacy at checkers. In fact, it really was a private contest of man versus man.

A second reason was to alleviate what I perceive to be a void in the literature. There are many interesting computer "story" books around, detailing such topics as the history of computing, companies, personalities, and even a few on technical products. But I don't know of any that go into the intimate details of creating a computer program, specifically an "intelligent" one. Tracy Kidder's *The Soul of a New Machine*, recounting the birth of a computer chip, was the closest model to what I wanted to achieve. In fact, *The Soul of an Intelligent Program* was a working title for this book at one time.

The third reason has to do with the educator in me. I hoped that I could write a lucid account of the trials and tribulations of developing a complex computer program. I wanted to educate people on the difficulties of writing computer software and give a realistic, understandable explanation of the latest computing technology.

The final reason is personal, and this is the one that I had to come to grips with. I feel a lot of dissatisfaction about the way the story ends. By forcing myself to express my feelings in words, it has given me the chance to contemplate what I've accomplished and, finally, allow me to restore my pride in the project.

There was a tremendous temptation to write a book that glorified the CHINOOK project, making it sound like a carefully planned, well-thought-out research effort. Of course, this is baloney. Instead I decided to write a book that tried to show the reality of research: personality conflicts, money, stupid errors, bad ideas, obsession, frustration, and the impact on the family. A friend of mine, on reading an early draft of the book, said, "You come across as a complete jerk." The early drafts were cruder with more personal "editorial comments" interspersed that didn't survive to the final edition. I tried to be honest, even to the point of saying some unflattering things about myself. I think this was necessary to give readers a flavor of what really went on. A "sanitized" version of the story wouldn't be as interesting and as informative. Nevertheless, I hope I don't come across as a jerk!

Apologies:

The book presents many things from my point of view and, unfortunately, often does not properly reflect the time and effort put in by other members of the team. Every one of them worked very hard and put in long hours. Please accept my sincere apologies if I have failed to acknowledge your contribution properly.

Before you read this book:

Computer people: Don't let the checkers-related material turn you off. Ignore it. Nevertheless, I encourage you to try to understand what's going on in the checkers diagrams or play through some of the games. You might discover that you actually understand the principles with minimal effort. Beware though: the game is addictive!

Checkers players: Don't dwell on the computer material. Ignore it. Nevertheless, I hope I have explained the concepts at a high enough level that you won't want to skip them!

For the record:

I had a serious dilemma in writing this book. Do you say checker player or checkers player? Logically, it made sense to call a player of checkers a checkers player. However, I quickly discovered that the checkers community (or is it checker community?) preferred the former. David Kramer did some research on this issue and wrote:

> After consulting Al Lyman, editor of the *ACF Bulletin*; a linguist at the University of Toronto; the world authority on billiards, a game with a similar plural problem; and nu-

> merous dictionaries and encyclopedias, I have concluded that in most cases it is in fact "checkers" that is to be preferred. Now, one moves a "checker" on the "checkerboard", but that's about it for the singular. However, according to Mr. Lyman...the plural [should be] reserved exclusively for "the game of checkers." I believe that a combination of linguistic forces has led to checkers players...making the "wrong" choice. These forces are euphony (it's easier to say "checker player") and what I would call guild snobbery, the attempt by the practitioners to elevate the language of their specialty above that of the masses...

For better or worse, I decided to stick with the "correct" usage.

Another departure from the checkers-community standards is in the representation of checkers moves. In this book I used the algebraic notation that is popular in the chess world. It is much easier to visualize moves using this notation than it is using the checkers community's numeric notation. Even though I've been using numeric notation for six years, I still find it awkward to use. Algebraic notation is simpler and, hopefully, will allow a wider audience to follow the checkers ideas.

The above points mean than some quotations have had minor consistency editing done to them. For example, checkers moves in the quotations specified using numeric notation have been changed to their equivalent algebraic representation.

All monetary amounts have been approximated to their U.S. dollar equivalents.

Despite numerous proofreads and double-checking of facts, it's inevitable that a few mistakes will be found in this book. All errors are my responsibility.

Finally:

Come visit our World Wide Web site and play a game against CHINOOK: http://www.cs.ualberta.ca/~chinook. Good luck!

January 1997 *Jonathan Schaeffer*

Acknowledgments (1997)

This book wouldn't have been possible without the important contributions of many people. I would like to extend my deepest appreciation to:

- CHINOOK team members Martin Bryant, Joe Culberson, Brent Knight, Robert Lake, Paul Lu, Duane Szafron, and Norman Treloar for their enormous commitment to the project;
- Marion Tinsley for bravely agreeing to play us;
- Derek Oldbury and Herschel Smith for their unswerving support; and
- Bob Bishop for giving us our fifteen minutes of fame.

Other people making important contributions to the development of CHINOOK include Brent Gorda, Jaap van den Herik, Randal Kornelson, Patrick Lee, David Levy, Steve Sutphen, and Ken Thompson. Thank you.

Support was received from the Natural Sciences and Engineering Research Council of Canada (NSERC), University of Alberta Central Research Fund, Department of Computing Science at the University of Alberta, Silicon Graphics International, the Netherlands Organization for Scientific Research (NWO), Computer Science Department at the University of Maastricht (formerly the University of Limburg), Lawrence Livermore National Laboratory (Eugene Brooks), and IBM.

The open-mindedness of the checkers-playing community gave CHINOOK its opportunity to compete against the best human players. I would like to thank the American Checker Federation, Marion Tinsley, Asa Long, Don Lafferty, and Derek Oldbury. All of them could have said "no" to a computer.

The contents of this book benefited from interactions with Martin Bryant, David Butler, Gil Dodgen, Eric Jensen, Brent Knight, Don Lafferty, Robert Lake, Paul Lu, Stephanie Schaeffer, Herschel Smith, Steve Sutphen, Norm Treloar, and Tom Truscott. Numerous people read the book and offered me valuable feedback: Martin Bryant, Jordan Devenport, Martin Devenport, Gil Dodgen, Dap Hartmann, Andreas Junghanns, Brent Knight, Richard Korf, Don Lafferty, Robert Lake, David Levy, Paul Lu, Aske Plaat, Stephanie Schaeffer, Manuela Schön, Steve Sutphen, Duane Szafron, Norman Treloar, Jaap van den Herik, and Judy Woken. In particular, Robert Lake, Aske Plaat, and Stephanie Schaeffer were brave enough to read the manuscript several times. Eila Smith suggested the title for the book. Robert Lake helped with the diagrams.

I would like to acknowledge three valuable sources of information: Mary Clark, Marion Tinsley's sister, for providing historical background on her brother; Donna Hussain, Arthur Samuel's daughter, for providing me with access to unpublished papers by Samuel (including his autobiography, *A Boy From Emporia*); and Jim Propp for his excellent article on the 1994 Tinsley-CHINOOK match (which, regrettably, was never published).

I would like to thank the people at Springer-Verlag: Martin Gilchrist, Victoria Evarretta, and Ken Dreyhaupt. In particular, Martin Gilchrist made this book a reality. David Kramer provided valuable feedback, rejuvenating my enthusiasm for the book just when it was lagging.

Contents

I Feel I Can't Lose

After making the first three moves of the checkers game, the arbiter, Con McCarrick of Ireland, reaches out and starts White's clock running. The White pieces are commanded by a tall, slim man dressed in a modest green suit, tie held in place by a clip with "Jesus" spelled out in colored stones. He pauses for a moment, makes his move, and then presses a button that stops his clock and starts Black's. His opponent, playing the Black side, immediately captures a piece. Just two friends playing a game of checkers, or so it seems.

There is a flurry of camera flashes as photographers jockey for position. After five minutes of this, McCarrick indicates that the time for picture taking is over; it's now time for the players to think. As the reporters withdraw, the field of view for the spectators widens, and they can see not just the adversaries, but a computer terminal perched beside the Black player. On a large screen overhead, a realistic-looking white hand occasionally reaches out and makes a move on a computer-generated board. Once complete, a sinister-looking black hand makes the next move. Back and forth the two hands move on the screen, mirroring the moves played between the two combatants. White and black, metaphors for good and evil. The careful observer notes that both hands have a wedding ring on the second finger from the right, and the second finger from the left is bent, as if it has been in an accident. Realism versus animation, metaphors for man and machine.

The White player, Dr. Marion Tinsley, is clearly the crowd favorite. He is a young-looking sixty-five years old. At the start of the game he is relaxed and smiling, confident about the outcome, seemingly oblivious to the obvious tension that fills the air. The day before, at the opening ceremony, Tinsley gave a speech in which he said, "A reporter over here said a while ago, 'You can't lose, can you?'...right now I am just free of all stress and strain because I *feel* I can't lose."[1] Dr. Tinsley is St. George and his opponent is the dragon. Confidence is a knight's greatest asset. Without it, all is lost from the start.

---

[1] Marion Tinsley quoted in *Silicon Graphics World Draughts Championship*, the daily bulletins for the 1992 World Man-Machine Checkers Championship.

Moving the Black pieces is Dr. Jonathan Schaeffer, whose appearance is unbefitting for his role as the dragon. At thirty-five years old with brown curly hair and looking uncomfortable in a suit jacket without a tie, he appears to be ill at ease playing the moves. While Tinsley exudes confidence, Schaeffer seems hesitant and unsure of himself. Each move is double-checked, as if he isn't sure how to play the game. In fact, Schaeffer is a novice player at best. Yet surprisingly, he's here playing for the highest honor in the world of checkers—the world championship.

If you watch the participants onstage closely, you notice that every time Tinsley makes a move, Schaeffer reaches over to a computer keyboard, types a few key strokes, and then stares, not at Tinsley or the checkerboard, but at the computer screen by his side. The screen is at an angle so that only Schaeffer can read the contents; Tinsley is forbidden to look at it. Sometimes Schaeffer's eyes wander off the screen and stare intently at his adversary. Tinsley pays no attention; he is lost in the world of checkers, as move sequences and checkers patterns mingle in his mind. Abruptly, something interrupts Schaeffer's stare, and after a quick glance at the computer screen, he reaches out and plays a move. Other than relaying moves to and from the computer screen, he hardly ever looks at the checkerboard.

On the other side of the room a large refrigerator-like box stands alone, shunned by everyone. Yet every time Schaeffer interacts with the keyboard a panel of lights on the box starts dancing, as if excited by the contact. If you listen closely you can hear the hum of fans inside, keeping the contents cool, and a noticeable clicking sound resembling that of a Geiger counter. Few in the room know that the "box" is a $300,000 state-of-the-art Silicon Graphics computer. Even fewer care that the machine actually contains eight computers, all working cooperatively to solve a problem: find the best move to play in the game.

It is August 17, 1992, and the venue is the five-star Park Lane Hotel in central London, England. Dr. Tinsley is defending his world championship title against the computer program CHINOOK. Dr. Schaeffer, a duffer checkers player by his own admission, is just babysitting the program. His role is solely to relay moves between the board, which is the battleground, and the computer.

In the physical domain machines have been superior to man for many decades. Would you want to run a race against a car? Would you want to compete against a forklift at weightlifting? But in the mental domain humans remain supreme. For the first time in history a computer has earned the right to play for a human world championship. Man, confident with his God-given gift of intelligence, is being challenged by a computer, a mere machine. Man the creator is being challenged by his offspring the computer.

Are we witnessing history in the making? Will the electronic computer master mankind, his maker? Can a computer win the world checkers championship?

The Opening Game

Chapter 1
This Was Going To Be Easy

"Jonathan, what ever happened to computer checkers?"

It was an innocent question, posed to me by my colleagues Joe Culberson and Duane Szafron. It was the winter of 1988, and I was in the middle of implementing a new idea in my computer chess program, PHOENIX. While taking a quick break for lunch, I ran into Joe and Duane. They stopped their conversation and, seeing the supposed expert on the topic being discussed, asked me the question. Everyone seemed to be working on programming chess. Why not checkers?

"Umm, let's see. There was this guy named Samuel, I think, who wrote a checkers program in the 1960s. It beat some strong player and, umm, eventually was good enough to compete with the best players in the world. Something like that anyway. No one cares about checkers. It's a kid's game. Why do you ask?"

It was a conversation that changed my life.

∞∞∞∞∞

Ever since I was a child, I enjoyed playing games. At age twelve I chanced to see two classmates playing chess, and I was hooked. From that day until I graduated from high school six years later, virtually every lunch hour was spent hunched over the chessboard. At fourteen I entered my first tournament, and two years later I was a candidate master. The beauty of the game had a hypnotic effect on me: the brutal intricacies of a deep sacrificial checkmating attack, with Arthurian pieces defending the honor of the king while simultaneously slashing at the heart of the enemy; the delicate subtlety of a few pieces weaving magic in a simplified endgame like a precisely scripted ballet. Just like a connoisseur of fine art or music, as my understanding of the game grew, so did my appreciation of its beauty. I treasured my collection of best games, wanting to frame each of them and put them on the wall, as if others could read the hand-scrawled sequence of moves on the game record and visualize the elegant interactions of the pieces as I could. I couldn't paint, and I had limited musical skills, but I could be an artist over the chessboard.

My parents and teachers were tolerant of my growing infatuation with the game. In reality, it was an addiction, but seemingly without any of the detrimental side effects. Chess allowed me to escape into a dream world without the monetary cost and physical damage of gambling, alcohol, or drugs. It allowed me to stretch the capabilities of my mind as I forced myself to sift through a maze of intricate complexities. As I began to solve some of the mysteries of the game, something began to happen to my school performance. When I started playing chess at the age of twelve, I was an indifferent student with average marks. Studying the game forced me to concentrate, analyze, use my imagination, and solve problems. Quickly, without any conscious effort, my academic performance rose until by the age of fifteen I became the top student at my school. Academic work seemed to become almost effortless. It was now easy to justify all the time I spent on chess, even if I did get labeled as a bookworm (fortunately, the word "nerd" hadn't yet been coined).[1]

In my last year of high school, my love of chess was gradually replaced: women entered my life. I tried playing in chess tournaments during the day, and then going out on a date at night. It didn't work; the games were on my mind during the date ("You're thinking about chess again, aren't you?" she'd say with a scowl), and the games the next day suffered because of my lack of sleep the night before. Something had to give. For the next five years I rarely played the game competitively, being preoccupied with my university studies and preferring to spend all my spare time pursuing my non-chessic queen.

In 1975 I went to the University of Toronto to study mathematics, but my teachers quickly turned me off the subject. I switched to physics, but found that the subject didn't come naturally to me. I was in a quandary, not knowing what subject to pursue. My computing course seemed easy, so in my second year I switched to computer science as my major. The next summer I stumbled across a new book in the university book store, *Computer Chess* by Monroe Newborn. I only had to read a few pages before I began thinking that I could write a computer program to play chess. When I got back to university in the fall I started to write one and, predictably, discovered it was much harder and more time-consuming than I had imagined. The chess program would have to wait.

While in my last year of studies at Toronto I began to think more and more about writing a chess program. I knew I could never be the world chess champion—I just wasn't good enough. But maybe, just maybe, I could become the world computer chess champion. I'm a fiercely competitive person, and the thought of becoming the best at something was enticing. It was obvious that building a world-class program would be a big undertaking, something that would be difficult to achieve in my spare time. I had an inspiration: do a master's degree and use computer chess for my thesis. I started at the University of Waterloo in January 1979.

My time at Waterloo greatly benefited from the presence of Ron Hansen. He was the author of RIBBIT (later called TREEFROG), one of the strongest chess programs

[1] Jonathan Berry writes in Toronto's *Globe and Mail* newspaper (August 26, 1995, p. 12): "Chess is fully 25% of the Challenging Mathematics curriculum in Quebec and New Brunswick, for grades two through six. A 1993 study reported that Quebec math scores were 15% ahead of the national average. A detailed study in New Brunswick revealed that, although their computational skills matched the control group before and after, participants in Challenging Mathematics showed 19% greater improvement in problem-solving and in problem comprehension after two years." See "The Benefits of Chess in Education", compiled by Patrick MacDonald, for a detailed summary of the research done on this topic (http://www.psmcd.net/otherfiles/BenefitsOfChessInEdScreen2.pdf).

around. He generously gave me a copy of RIBBIT, which I used to learn how to write a chess program. Programmers write their programs using a language specially designed to cater to the computer's limited capabilities. Hansen's program was written in a programming language called FORTRAN. As part of my master's thesis, I translated it into a more-modern programming language. This forced me to understand every intricate detail, in much the same way someone would become intimate with Shakespeare's writing if they had to translate his collected works from, say, English into French. Now that I had a complete program that I understood, it was time to conquer the world. It was time for me to add my expertise to the program and create the ultimate chess player. It was time to beat the world champion!

Everything I read about chess programs convinced me that they were ignorant; they had little in the way of chess knowledge. Of course, since I knew a lot about chess, it would be a simple matter of translating my expertise into code and, voilá, success! I spent a year working on the program, adding as much knowledge as I could to it. The new program, dubbed PLANNER, failed to live up to my performance expectations. Gradually my enthusiasm began to wane. The chess knowledge that I added was simple because important concepts seemed hard to program. The machine required a precise specification, but my chess knowledge was imprecise. Further, for every piece of knowledge that I added, there always seemed to be an endless stream of exceptions. This was going to be harder than I thought.

I finished my master's thesis, titled *Long Range Planning in Computer Chess*, and graduated in 1980. I didn't know what to do next, but my supervisor, Morven Gentleman, whispered the magic words "Ph.D." to me one day, and my ego wouldn't let go of the thought. I hadn't really considered the possibility, but the mere suggestion set my head spinning. The thought of being called *Doctor* Schaeffer was intoxicating. Hey, I could spend four years playing with my chess program and get a Ph.D. out of it. Not bad for a kid who didn't want to grow up. I carefully weighed the alternatives, or so I told all my friends and family, and chose to remain a student. I went to discuss my decision with my supervisor. He warned me that computer chess work would lead nowhere. "Do a Ph.D. in something respectable," he said. I walked out of his office in a daze. Checkmate.

Never underestimate the motivating force of the ego. Maybe I wanted to impress my family (my grandfather always wanted me to be a doctor). Maybe I thought I could fool everyone and disguise my computer chess intentions. Whatever the real reason, I know in my heart that my ego played a decisive role in my decision. So in September 1980 I started the Ph.D. program at Waterloo, working in the area of parallel computing, and I publicly swore off computer chess. I had to get serious about a thesis "in something respectable." My firm resolution lasted eight months.

If I was going to create the world champion chess program, then I would need help. I advertised my project within Waterloo's Department of Computer Science and was fortunate to find Howard Johnson, a fellow Ph.D. student, who was as enthusiastic about computer chess as I was. The summer of 1981 was spent writing a new program that we called PRODIGY. Howard wrote the control part of the program, and I put in the chess knowledge. We entered it in the 1981 North American Computer Chess Championship. Against the best programs in the world, we fared poorly. The program exhibited moments of brilliance, only to come crashing down in every contest. We lost all our games and finished dead last. I was bitterly disappointed. My enthusiasm for computer chess disappeared abruptly on the last day of the tournament, and PRODIGY never played again.

Over the previous seven years I had hardly played any chess. Occasionally I consented to play in a tournament, mostly for the social contact rather than the competitive spirit. Although I wasn't studying chess, working on a program forced me to think about the game and its inner structure. I must have been absorbing something from this experience because I suddenly found my chess rating soaring. I became a master and one of the top fifty players in Canada.

My Ph.D. wasn't going well, so in the summer of 1982 I started looking for a distraction. Yes, I started writing yet another chess program, this one called PHOENIX (it rose from the ashes of PRODIGY).[2] The PLANNER and PRODIGY experiences were invaluable as they convinced me that, contrary to all my expectations, lots of chess knowledge didn't work. Which programs were winning the tournaments? The ones with little knowledge but with the ability to consider an enormous number of chess positions. With a twinge of regret, I wrote PHOENIX to mimic these "dumb" programs. The results were immediate. PHOENIX didn't know nearly as much about chess as PRODIGY did, but it would continually beat it game after game. Obviously, my old approach, imparting human knowledge to an inanimate machine, wasn't the best way to train a computer to play strong chess.

PHOENIX qualified to participate in the 1983 World Computer Chess Championship in New York. The program played well, winning two games, losing two, and drawing one. The final result was creditable and showed that PHOENIX wasn't far behind the best programs. My "yo-yo" years of computer chess, up and down, were on the upswing again. I knew I could do it. I could be a world champion.

All this computer chess time didn't come for free. My Ph.D. thesis was in a shambles because I wasn't devoting enough time to my research. Discouraged, I decided to quit school, get a job, and see what life was like with a real salary. I sent out my résumé and had a few job interviews. None of the employment offers that I received appealed to me. Now I was discouraged about my job prospects. Just in the nick of time, a friend, Randy Goebel, suggested I take my work in computer chess and turn it into a thesis. After all, he argued, the most important thing was getting the degree; the research topic was irrelevant. A job only had the attraction of money, but the chance to turn my chess work into a thesis seemed irresistible. At the invitation of Tony Marsland, one of the major players on the computer chess scene, I moved to the University of Alberta in Edmonton, to complete my degree. He arranged for me to teach as a lecturer at the university while I worked on my thesis part-time. By mid-1985 the thesis was done, although I didn't graduate until 1986. The thesis, *Experiments in Search and Knowledge*, became an important work

---

[2] Even if I wanted to, I couldn't use the name PRODIGY again. Six months after the North American championship I was startled to see an advertisement for a chess computer named PRODIGY. I wrote to the manufacturer asserting my prior claim to the name. They wrote back stating that they had done a trademark search on the name and found no matches. Therefore, they would appreciate it if I would stop using their name. They left no doubt about the legal implications of their request.

in the area. I became an Assistant Professor at the University of Alberta starting in September 1985.

As a professor I was free to research whatever I wanted as long as I produced scientific papers. What a deal! I could work full-time on my chess program and get paid to do it. Surely this was the ultimate job.

I worked hard on PHOENIX in preparation for the triennial World Computer Chess Championship in 1986. To improve the program's performance it was modified to run in parallel, using up to thirty computers. They would divide up the work, and each computer would solve part of the problem. In effect, the program was like a small business organization, with a manager to allocate work and employees to do the assigned tasks. The hard work paid off; PHOENIX tied for first in the world championship. I partied late into the night after the final game, intoxicated with success and Coca-Cola. It took a long time for me to come down from my high.

Although work continued on PHOENIX for the next few years, progress was slow. The problems I wanted to solve seemed to get harder and the solutions more elusive. By the time the 1989 World Computer Chess Championship was in sight, I was working hard on the program but without the enthusiasm of previous years. My major competitors had access to more money, personnel, time, and computing resources than I did. To make matters worse, IBM was about to hire the new World Computer Chess Championship team DEEP THOUGHT (soon to be renamed as DEEP BLUE). I couldn't compete with IBM's deep (pun intended) pockets.

I didn't enjoy seeing PHOENIX becoming an also-ran—I was in this to win (and, of course, to do some research). Preparing for competitions was no longer fun. It was hard work. Hard work without the satisfaction of winning.

And then came the fateful day that I went to lunch with Joe and Duane.

∞∞∞∞

Why would anyone want to write a game-playing program? Why would I undergo all the hard work, long hours, and frustration, just to create a mere computer game? Imagine the excitement of creating life out of nothing, much like Dr. Frankenstein did. Okay, so it isn't quite as dramatic, but the idea is still the same. You have a computer that is like an empty mind, devoid of intelligence. All you have to do is teach it! With infinite patience one gradually refines one's teaching skills and bestows upon the computer the semblance of intelligent behavior.

I vividly recall how startled I was the first time PHOENIX almost defeated me in a chess game. I probably wasn't paying too much attention to the game since I'd never lost to the program (I needed an excuse). One careless mistake and—oops—I was in a lot of trouble. Squirm as I might, I couldn't free myself from its python-like grip. With defeat staring me in the face I did the only reasonable thing: I turned the computer off. It was an "accident," of course. Gee, I guess I win on forfeit now.

That game shook me up. In many ways it was the realization of a dream. I was a master player and yet had almost lost to my creation, like a father losing to his son for the first time. I was exhilarated at the success of my work and yet felt terribly vulnerable. I was a human being, after all, and felt uneasy about a computer program of my own creation that might soon be my superior. I could create intelligent behavior. I could create a chess monster. I was scared.

For most people chess is just a challenging intellectual game to be enjoyed. Until I went to Waterloo, chess was a fun pastime. Now, as a professional computer scientist, I had to view chess as an application domain for my research into artificial intelligence, making computers exhibit intelligent behavior. It was still fun sometimes, but now that it was part of my job, it was also work.

As a chess player I only understood the game at an artistic level, trying to uncover the analytic truth and beauty of each position. As a scientist I had to try to understand it at the cognitive level. I could play chess well; what would it take to make a computer play as well as I did? This led to a deep philosophical question: if computers could play chess better than any human, would they be "intelligent"? Thus, my humble work on computer games was really designed to give me insight into intelligence—man and machine.

The complexity of human intelligence is currently beyond comprehension. For example, just the process of understanding the text you are now reading requires an enormous amount of knowledge. We're able to read a book with ease, yet the same task is overwhelmingly difficult for a computer. Science fiction writers tell stories about computers that are superior in intelligence to man. In reality, that day is many decades away, and it likely won't occur in my lifetime. The challenge of creating computer intelligence is so complex that we need to start with something simple and then progress. If you want to write a computer program to do something intelligent (like reading a book), you should start with a small task (like reading limited-vocabulary children's books). If you succeed with that problem, then move on to something more challenging. You have to learn to walk before you can run. And so it is with computer games.

Games such as chess and checkers are ideal domains for exploring the capabilities of computational intelligence. The rules are fixed, the scope of the problem is constrained to a small 8×8 square battlefield, and the interactions of the players (pieces) are well defined. Contrast this to the real world—the game of life—where the rules often change, the scope of the problem is almost limitless, and the participants interact in an infinite number of ways. Yes, the games problem domain is vastly simpler than the complexity of life that you and I face every day. Surely though, if you can't create an intelligent program in such a "simple" domain as chess or checkers, how can you hope to succeed with more complex problems?

For a program to play a strong game of chess or checkers, it must be able to do many things that humans do. For example, it must be able to analyze positions, searching among the possible moves to decide which one is the best. It must have knowledge to differentiate good from bad, and learn so that it doesn't make the same mistake twice. All of these things are characteristics we ascribe to human intelligence, and somehow they must be created in a computer. A daunting task.

Have you ever tried to understand how your own thinking process works? Make a decision and then try to analyze the method you went through to arrive at that decision. You'll probably have no idea what happened. Sometimes you're able to relate a sequence of thoughts that logically leads to the decision you just made, but most of the time you can't—it's too difficult. Usually, the decision just pops into your head, and you don't know how it got there. Magic. For many hours I would lie down with my eyes closed trying to figure out what was going on in my head. It all proved futile. The brain hides its secrets well. And yet, I hoped to create a program that mimics this behavior. As I write these words I try to understand where they are coming from. How do I decide what to write? How do I logically organize my thoughts into coherent sentences? The words readily flow from my mind to my fingers typing on the keyboard. It's like a waterfall, except that I can't see the source of the water.

Watching a child grow up makes you aware of how little we know about intelligence. When my daughter was born in 1991, I was determined to study her, hoping to learn some of the secrets of intelligence. One day she started talking. How did she do that? How could she learn all those words, their meanings, and the grammar necessary to string an intelligent sentence together? It seemed to happen overnight. Computer scientists have been working for decades at programming computers to understand English, yet the skill level of the computer isn't as good as that of my daughter when she was two years old.

The more you work with computers trying to create intelligent behavior, the more you realize what a marvelous creation the human mind is. Things that we take for granted, like speech, vision, problem solving, and learning, are incredibly complex processes. It's amazing that we can carry out these tasks seemingly without effort. Computers can do simple tasks such as addition and multiplication faster and more precisely than can humans. For the complex tasks that we consider to be the hallmark of intelligence, the brain remains superior.

Although the above points sound good to the layman, to my scientific colleagues these arguments are just fluff. From the academic point of view, one needs compelling justification for the scientific merit of building a strong game-playing program. For example, Professor Alan Bierman writes,

> It would seem, after all, that the major goals of computer science are to discover how to increase the capabilities of machines and that the domain of games offers a perfect laboratory for studying complex problem solving behavior. We would like machines to help us solve problems in many areas, business, government, medical, scientific, legal and others, but in each of these areas we find it extremely difficult to build into a machine a world model that is complete enough and accurate enough to enable it to do non-trivial decision making of the type that humans do routinely. Also in these applied areas, it is not necessarily easy to judge whether the decision maker, man or machine, is making good decisions because there are few commonly accepted measures of goodness. On the other hand, in a game like checkers or chess, the machine can hold and properly model all of the relevant information about the particular game and the measure of the quality of the behavior is absolute. One simply plays it against a competitor and observes whether or not it can win the game. It's also quite fortunate that there are a number of human experts in each of these games so that we can study their performance in contrast to that of machines and learn a little more about both.[3]

Sounds convincing to me. Alan, can you help write my research grant proposals?

If I moved my research application domain from chess to checkers, I would be able to address the same fundamental research problems I was addressing in chess,

---

[3] Alan Bierman, "Theoretical Issues Related to Computer Game Playing Programs," *Personal Computing*, September, 1978.

but in the simpler domain of checkers. However, I knew little about the game, other than the rules and the experience of playing a few games as a child. So why even bother with checkers when I obviously had everything going for me with my chess program? Well, I could give you a lot of valid scientific reasons, but deep in my heart I knew the true answer: I could win. There seemed to be a scientific void: in the rush to build strong chess programs, the scientific community had neglected checkers. Here was a chance to fill the void and achieve something no one had yet achieved: build a program capable of beating the human world champion. In chess this possibility was slipping away from me. But checkers offered me the opportunity to achieve a laudable scientific goal, albeit with a different game. Hmm. I wondered whether my computer-chess friends would regard me as a defector.

There was one intriguing aspect of checkers that immediately attracted my attention: checkers was a "simple" enough game so that maybe, just maybe, it was even possible to solve it. Tic-tac-toe is a solved game; everyone knows that unless someone makes a mistake then every game will end in a draw. Was it possible to do the same thing with checkers? Solving the game means always playing the right move in every position with no tolerance for error.[4] I wondered if we could do it. Accomplishing that would be quite a coup.

Solving checkers is a different problem from trying to build a world championship program. Of course, if you build a perfect checkers player then you must also be the best player in the world. Building a strong game-playing program is something I knew how to do: you use knowledge to search for the move most likely to improve your position. I knew little about how to go about solving a game. You need to know whether each position is a provable win, loss, or draw. You aren't concerned with trying to find the best move to play; you *must* know what the best move is. In effect, solving the game is a much harder problem.

What would it take to solve checkers? I wasn't sure, but I did know something about the difficulty of solving chess. Various people have estimated that there are something like 1,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000 possible chess positions, or, more succinctly, 10^{45}. Of course, this is an astronomically large number (in the same ballpark as the number of atoms in the universe). But checkers is a simpler game: only thirty-two squares are used on the board, as opposed to sixty-four for chess, and there are only two piece types (king and checker), compared to six for chess (king, queen, rook, bishop, knight, and pawn). How many possible checkers positions are there? Joe Culberson figured out the answer: 500,995,484,682,338,672,639. In other words, roughly five-hundred billion billion, or 5×10^{20}. To put a number this big into perspective, imagine the surface of all the earth's land mass as being equivalent to the number of possible checkers positions. Then one position is roughly equivalent to one thousandth of a square

[4] There are actually several definitions of "solving" and this will be discussed in more detail in Chapter 24. The simplest one is knowing the result of a game played if neither player makes a mistake (knowing that tic-tac-toe, in the absence of errors, is a draw). A harder problem is to be able to play a game without making a mistake (never lose a game of tic-tac-toe). The hardest problem is to always make the "best" moves—never make a mistake, and always select your move to maximize the likelihood of the opponent erring (increase your chances of winning at tic-tac-toe).

inch. Alternatively, pretend that the Pacific Ocean is empty, and you have to fill it using one small cup. The number of cupfuls of water that you pour is roughly the same as the number of checkers positions.

Actually, the number of positions calculated by Joe was overstated. What he computed was the number of different ways of placing up to twelve pieces (kings or checkers) for each of two colors (White and Black) on the checkerboard. However, some of those positions can't arise in a game. For example, take the starting position and replace one of the white/black checkers with a white/black king; there is no way this position can be reached by a legal sequence of moves. We couldn't figure out any way of excluding these unreachable positions.[5]

From an innocent chance encounter at lunch my interest in checkers had been piqued. More importantly, Joe and Duane were also intrigued. With naive dreams of possibly solving the game, Joe, Duane, and I began meeting for an hour every week to discuss the problem. Initially, we decided to figure out how many of the possible positions with twenty-four pieces on the board were legally reachable from the start of the game. There are ninety billion billion positions with twenty-four men on the board, but most must be illegal. So I wrote a program to try to calculate how many were legal. In effect, the code I wrote was a simple checkers program. In each position it figured out which moves were legal, played one of the moves on the board, and then examined the resulting position. The program would only consider positions with twenty-four pieces on the board; when it reached a position with a capture move, which would reduce the number of pieces below twenty-four, the program would move on to another position. Initially I thought there might be only a few thousand legal positions with twenty-four pieces on the board. I let the program run for an hour, and it discovered a few million positions before I stopped it. Obviously, the game was more complicated than we thought.

The weekly meetings continued, each generating a fresh set of ideas for conquering checkers. Many led to computer implementations to test the idea's feasibility. Usually, the computer results were discouraging; the problem was too big. Gradually, a rather sophisticated program was built that was useful for experimenting with checkers. Unfortunately, it wasn't designed to play a complete game.

In April 1989 work on checkers stopped. I was busy organizing (with Tony Marsland) the World Computer Chess Championship in Edmonton, and the tournament was rapidly approaching. Everything—checkers, graduate students, love, and life—was put on hold as I divided my time between making this large event a reality and trying to get PHOENIX ready. The last two weeks before the start of the event seemed to be one long sleepless day as I worked on the myriad of last minute organizational details during the day and applied bandages to PHOENIX at night.

PHOENIX played in a strong human tournament in Vancouver the weekend before the start of the world championship. The results were encouraging and at the same time disappointing. The program defeated two strong masters, both players ranked in the top fifty in Canada. This was quite a triumph; they were the strongest

---

[5] In the first edition, I wrote that "...it's impossible from the starting position to construct a series of moves leading to twenty-four kings on the board." Wrong! Thanks to Richard Pask for pointing out that this has been known for over a century.

players PHOENIX had ever beaten. But on the down side, PHOENIX had several strong positions that it let get away and, indeed, even lost one. Also, the program was crushed by a player who wasn't even in PHOENIX's league. How could the program defeat one of the best players in Canada in the morning and then be outplayed like a child by a much weaker player in the afternoon? The program should have been named ENIGMA. Nice name—for my next chess program.

The world championship was held May 28-31, 1989. My worst fears were realized and PHOENIX was a disappointment. It repeatedly snatched a draw from the jaws of victory, finishing in the middle of the pack. Preparing for this event took considerable effort and, in the end, was unrewarding. The fun was seemingly gone; developing the program was hard work. When the fun goes, so does the motivation. In a moment of weakness I supposedly said, "I'm finished with computer chess after this championship." Did I really say that? My girlfriend, Steph, claimed I did and that I "promised to spend quality time with her." She kept reminding me of those statements every day of the tournament. The event was over, and a decade of work on computer chess seemed to be coming to an end.

It may seem strange that I could just abandon the passion of the last ten years of my life so easily. In truth, the last year or two hadn't gone well. Every new idea I tried with the program seemed to go nowhere, and I had to invest lots of effort to achieve those negative results. Even when success came my way, the gains were modest at best. I felt like I had exhausted all my good ideas, and new ones weren't popping into my head as often as they once had. I recall many nights when I couldn't sleep because my brain wouldn't turn itself off. Ideas would suddenly materialize and hours of feverish sleepless activity would resolve all the unanswered questions of the new idea. Often I was so excited that I would get out of bed in the middle of the night and rush off to work. If I had a five o'clock shadow on my face at nine o'clock in the morning, it meant I had arrived at work before five o'clock in the morning. Those sleepless nights often led to my best ideas, but they weren't happening any more. I was sleeping well, and that was very disturbing.

My work was at a crossroads. The reality was obvious—stop working on computer chess—but the alternative wasn't. Could one stop, just like that? Or was computer chess like smoking cigarettes: you never really get rid of the craving? There have been many times when I overdosed on my work, but in every case a few weeks of mental diversion was sufficient to refresh me and reinvigorate my life. Things seemed different now. I had given PHOENIX my best effort over the past year, but it had been largely unproductive and the results showed it. Was this just a valley, a prelude to a peak on the horizon, or was I burnt out? For the first time in my life I believed it was the latter. I remember reading that physicists produce their best research before they are thirty-five years old. After that it was almost always downhill. I was almost thirty-two years old. Maybe my best years were behind me. The self-doubt had been building for months, but only during the world championship did I start worrying about it. The tournament was over. Now what would I do?

Yes, it is cliché, but fate does work in mysterious ways. In my life there have been several strange coincidences that seem to defy explanation. It's almost as if fate was intervening at a timely moment. I recall being a lost Ph.D. student at Waterloo, thinking of quitting. A timely e-mail from Tony Marsland, and suddenly I had a new job and my thesis, a new lease on life. Fate? Coincidence? Who knows and who cares. It all worked out in the end. And now, again at a crossroads in my life, fate chose to intervene.

David Levy, the president of the International Computer Chess Association (ICCA), was in Edmonton for the world championship and was advertising his forthcoming Computer Olympiad. The Olympiad, to be held in August 1989 in London, was a computer-only competition in fourteen games, including backgammon, bridge, checkers, and chess. The idea was to mimic the human Olympiads and award gold, silver, and bronze medals to competitors in a number of events. Whereas the human Olympiads were exhibitions of human physical skills, the Computer Olympiad was to be an exhibition of computer intelligence.

Working on chess so soon after the world championship seemed unthinkable, especially after my promise to Steph. There seemed to be only one reasonable thing to do: turn the checkers experiments into a functional game-playing program and enter the Olympiad (and get a trip to London in the process). I made up my mind the day after the world championship ended. It didn't take much persuasion to get Joe and Duane to commit to the project. Work immediately began on the checkers program, and with it I began to log the major events along the way.

Programmer's log, checkers project, day 1
Friday, June 2, 1989

Steph wants to know why I'm working on the computer at home. "The world championship is over. Relax. Let's take a holiday." Oops. A roadblock already. I can't recall my exact words to her, but whatever they were I must have been at my oratorical best. I am charming, understanding, sympathetic, and patient. At least that's my side of the story. Her view is slightly different: I am dishonest, insensitive, self-centered, and, to be quite blunt, a jerk.[6] Somehow, I manage to postpone my "quality time" with her until September, after the London Olympiad. Sometimes I amaze myself at my own stupidity.

Work starts on the program a mere two days after the world championship ends (I need a day to catch up on my sleep and give Steph a chance to calm down). June 2, 1989 becomes the first day of our quest to become world checkers champion. The clock is now ticking.

∞∞∞∞

[6] Here is an appalling example of the liberties taken by an unscrupulous author (author name and book title withheld). I was horrified when I saw the text from this book taken out of context (p. 62): "A diary [Schaeffer] kept while working on CHINOOK reveals that he was aware of his failings—'I am dishonest, insensitive, self-centered, and, to be quite blunt, a jerk'—but Schaeffer's task took precedence over everything else." I am at a loss to describe this deliberate act of misrepresentation. Steph was very angry: "He had to steal my line and give you the credit. What a jerk!" Uh, wait a minute. I'm not sure I came out so well in that exchange.