-- A GAMMON PRESS ANNOTATED MATCH --

Learning from the Machine: Bill Robertie versus TD-Gammon

by

Bill Robertie

$20

Table of Contents

Match Summary

Game	Result		Cumulative Total		Moves
	Robertie	TD-Gammon	Robertie	TD-Gammon	
1		2		+2	20
2		1		+3	17
3	2			+1	22
4	1		=		21
5	1		+1		9
6	4*		+5		39
7	1		+6		7
8	2*		+8		15
9		1	+7		10
10	1		+8		7
11	1		+9		8
12		1	+8		7
13	2		+10		31
14	4g		+14		27
15	1		+15		6
16	8***		+23		36
17	1		+24		20
18		1	+23		16
19		2	+21		23
20		1	+20		7
21		1	+19		6
22		2	+17		24
23	4*		+21		24
24	2*		+23		32
25		1	+22		7
26		1	+21		7
27		4**	+17		23
28	2		+19		22
29		1	+18		6
30	2		+20		33
31		1	+19		11

Key to "Results":

g = gammon won.
bg = backgammon won.
** = redouble occurred.*

Introduction

In the fall of 1991, I got a call from Dr. Gerald Tesauro, at the IBM Research Labs in White Plains, New York. He told me that he had finished work on his latest creation, a backgammon program called *TD-Gammon*, based on neural network theory. The program had been under development for some time, and had surpassed his previous effort, a backgammon program called *NeuroGammon*. Would I be interested in coming to White Plains and playing a few games against the machine, as a real-world test? "Sure", I said.

NEURAL NETWORKS

For those unfamiliar with the term, neural networks are the latest, hottest development in artificial intelligence software.

Traditional artificial intelligence programming requires the programmer to reduce the knowledge of a human expert in some field of activity to a set of heuristic rules, which are then incorporated into a program. If the rules accurately capture the human's knowledge, the program should be able to reproduce human expertise. For a few problems, like medical diagnosis or geological exploration, this approach has worked well. For the most part, however, the results have been disappointing. What a human expert "knows" about a subject is often subtle and complex, and the process of trying to reduce this knowledge to a manageable set of rules usually produces something less than expert behavior.

Enter neural networks, which attack the problem from the other end. If you can't teach a dumb machine to behave like a smart human, why not build a smart machine? Neural networks are essentially model "brains" built out of "neurons" created either in hardware (fast, but expensive) or software (slow, but cheap). Like neurons in the brain, each machine neuron has a series of inputs and outputs, corresponding to synapses in the brain. The programmer defines a problem (playing good backgammon, for example), and figures out a way to represent the problem (the backgammon board, and the rules of legal play) in the program. The program then plays backgammon with itself, and tries to "learn" from its mistakes.

That was the theory, anyway. I was certainly intrigued to see how far *TD-Gammon* had come. Was it a duffer, like most programs? Had it learned all that we humans had learned in 2,500 years of playing this game? Had it (shudder) surpassed us? Was I walking into a lion's den? Could a carbon-based life form overcome the silicon behemoth?

GAMES AND COMPUTERS

The history of computer game-playing began in the 1950s, when the Defense Department, through its Advanced Projects Research Agency (DARPA) began funding small-scale research projects involving chess programming. The motivation was straightforward: chess was seen as a simplified domain for various sorts of real-world problems; learning how to program a computer to play strong chess would presumably lead to the development of computers that could solve tactical battlefield problems in real time.

The early chess efforts were crude. In some cases, the chessboard was simplified to a 6 by 6 board to make the programming easier. The programmers attempted to reduce human chess-playing to several rules-of-thumb, which were then incorporated into the code. The resulting programs were easily beatable by any reasonably adept human. The first notable success in game-playing programming came not in chess but in checkers. In the late 1950s, a group at Carnegie-Mellon developed a checker program with some learning features which was able to play on a par with human checker experts.

The first real breakthrough in chess programming came in 1966, when Richard Greenblatt and a team of programmers and chess masters at M.I.T. created *MacHack*, the first program to play in chess tournaments and earn a chess rating. The program eventually achieved a rating of around 1500, about average for human tournament players. Although the program adopted a far more sophisticated set of rules and principles than early programs, it still played a weak strategical game.

The next, crucial breakthrough came in the 1970s. Computers were becoming faster and faster, as well as cheaper and cheaper. Soon programmers realized that it was now possible to write chess programs which largely dispensed with the heuristic rules that had guided earlier efforts, but which instead looked at all possible moves for some considerable depth into the future. Such a program *couldn't* make a tactical mistake within the limits of its search. As computers got faster, the programs would be able to look farther ahead within the same time frame and thus play even better.

The results were astonishing, even to computer optimists. By the late 1970s, chess programs had reached master strength. By the mid-1980s, senior master strength. By 1990, the best programs, such as *Deep Thought*, were beating grandmasters with regularity. By the end of this decade, a computer should be a favorite even against the human World Champion.

BACKGAMMON COMPUTERS

Backgammon programming began with the backgammon craze which swept the country in the 1970s. The very first efforts, mostly running on the early personal computers, were crude. However, unlike the case in chess, a breakthrough came very quickly.

Dr. Hans Berliner, a chess master and former World Correspondence Chess Champion, decided to use backgammon as a laboratory for developing some programming techniques for chess programming. The result, unveiled in 1979 and described in a Scientific American cover story in June 1980, was *Gammonoid*. The program was used by Black & White Scotch and Merit cigarettes as a promotional vehicle, and played an exhibition match against Luigi Villa after he won the World Championship in 1979. (*Gammonoid* won, 7-1).

Gammonoid was by far the best backgammon program of its time, about equivalent to a very strong intermediate or weak advanced player of the period. It was, however, lucky to beat Villa, who outplayed the machine. Berliner went on to develop the chess program HiTech, which was the best chess program until surpassed by Deep Thought (developed by Berliner's pupils) in the late 1980s. The *Gammonoid* program was available for play for many years to anyone who had use of the ARPANet network. Attempts to commercialize it, however, never materialized.

TD-GAMMON

TD-Gammon runs on an RS-6000 workstation, a considerably more powerful machine that even the fastest 486-based PCs that are now available. In the beginning, it was programmed only with the actual rules of backgammon and the ability to generate legal moves. No knowledge was programmed into it as to what constituted a good or a bad move. It "knew" nothing about making points, hitting blots, or anything else. Over the course of several months, it played 300,000 games against itself. In the beginning, it picked its moves at random from the list of possible legal moves. After each game, a logic routine made informed guesses as to what moves in the previous game may or may not have been errors, based upon a sophisticated mathematical theory of learning. (Keep in mind that the purpose of developing *TD-Gammon* was not to write the world's best backgammon program, but to use backgammon as a suitable vehicle for testing the validity of this learning theory.) The program's positional evaluator routine continuously modified itself, based upon its results in previous games.

After 300,000 games, the program had made considerable progress. But how would it do against humans? On October 22, 1991, I flew to White Plains to play against the machine. Was I impressed? You bet.

TD-Gammon and I played most of a day, a total of 31 games. I won 19 points, an average of 0.61 points per game. On balance, I was lucky. Game 16, as you will see could easily have gone the other way at the end, a 16 or 24-point swing by itself. My estimate, after reviewing the entire session, was that I would do well to average 0.20 to 0.25 points per game. This figure, if accurate, would make *TD-Gammon* the strongest backgammon program ever, better than Berliner's program from 1979.

Not only is *TD-Gammon* interesting as a backgammon program, it represents an astonishing achievement for the neural network approach to artificial intelligence. Remember that this program has no human knowledge built into it. Everything it "knows", it deduced by playing against itself, then modifying its approach after each game.

— — — — — — — — — — — — —

A word about the commentary. For the most part, these are my own notes. However, *TD-Gammon* also has an analysis capability, and for a few key moves I've included *TD-Gammon*'s analysis (given in italics). *TD-Gammon* can evaluate plays in three ways:

. ***One-Ply Evaluation:*** This is just the program's static evaluation of the position after that move, given by its internal evaluation routine. The move with the highest score for a one-ply evaluation is the move *TD-Gammon* will make in practice.

. ***Two-Ply Evaluation:*** This is the program's evaluation after taking into account all of the opponent's possible responses to each play, weighted by their frequency. In effect, the program is looking one move ahead. This evaluation is not available during play. This is more accurate than one-ply evaluation.

. ***Rollouts:*** *TD-Gammon* can also roll a position out any number of times and report on its results. This is its most accurate evaluation, but is time-consuming.

The numbers in *TD-Gammon*'s evaluations indicate the average points expected to be won or lost after each play.

— — — — — — — — — — — — —

In all diagrams, I have the White pieces (bearing off at the upper left) and *TD-Gammon* has the Black pieces (bearing off in the lower left).