

able to converge the parameter vector to an optimal state, we will be able to use temporal difference algorithms more effectively and in a wider variety of applications.

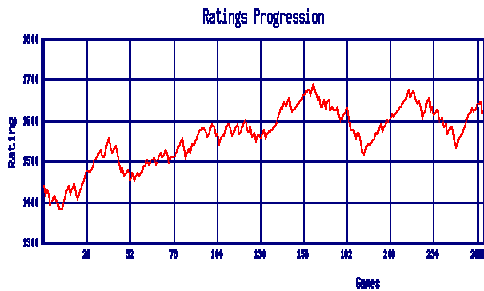


Figure 2: Ratings Progression Beginning with a Material only Vector

Initial data, though showing TDLeaf to be effective, suggest limits to the algorithm's ability to adjust the evaluation vector to an optimal state. Figure 2 plots the ratings progression beginning from a vector in which only the values of the pieces are nonzero. To get a baseline, the engine first played online (against human opponents) with the material-only vector without learning. After 340 games the engine's mean rating was 1504. A separate run of 265 games with learning on (shown in Figure 2) ended with a mean rating of 1558 – substantially higher than the previous non-learning run. The results of a subsequent experiment were, unfortunately, not as promising. Figure 3 illustrates the progression of a separate run of games that was started using a hand-tuned vector of evaluation parameters. After another 265 games the engine had achieved a mean rating of 1715. This is substantially lower than the 1775 mean rating from a previous run of a non-learning engine using the same hand-tuned vector. In this case, the learning was not only ineffective, but was harmful. The vector was moving away from optimal.

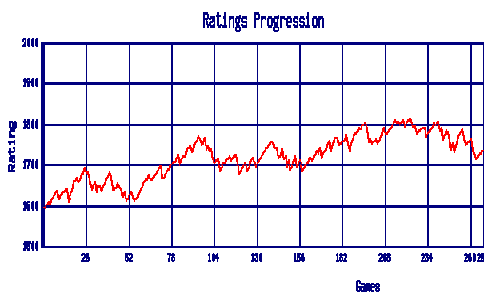


Figure 3: Ratings Progression Beginning with a Hand-Tuned Vector

Copyright © 2002, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

There are a number of conditions that may delay or even prevent convergence. Shallow searches, for example, leave the program vulnerable to the horizon effect, eventually causing the program to lose for tactical rather than positional reasons. In these cases it is likely the TD algorithm is adjusting the vector to predict tactical blunders, and consequently limiting the effectiveness of the program's ability to predict outcome based strictly on positional features. The idea that search depth adversely affects the algorithm's ability to converge to optimality can be tested by running several series of training sessions, each session beginning with an engine that plays at a higher caliber than the last, using the hand tuned vector. Evaluation complexity, distance from the optimal vector, quality of opponents, or length of training games may also delay convergence. Tesauro reported continually improving performance with TD-Gammon after hundreds of thousands of games (Tesauro 1995). Other dynamic or nondeterministic aspects of the program may also confound the algorithm's ability to converge to optimality. Examples include null-move forward pruning, aspiration windows, and principal variation search.

Acknowledgements

The author wishes to acknowledge Dr. Ronnie Smith and Dr. Mike Spurr of East Carolina University for their guidance and helpful insight, and his wife Amy for her support and unending patience.

References

- Beal, D. 1997. Learning Piece Values Using Temporal Differences. *International Computer Chess Association Journal*, 20:147-151
- Baxter, J., Tridgell, A., and Weaver, L. 2000. Learning to Play Chess using Temporal Differences, in *Machine Learning*, 40:243-263.
- Sutton, R. 1998. *Learning to Predict by the Methods of Temporal Differences*, Boston, Kluwer Academic Publishers
- Schaeffer J., Hlynka M., and Jussila V. 2001. Temporal Difference Learning Applied to a High-Performance Game-Playing Program, in *Proceedings of the 2001 International Joint Conference on Artificial Intelligence (IJCAI-2001)*, 529-534.
- De Jong, K. and Schultz, Alan C. 1988. Using Experience-Based Learning in Game Playing, in *Proceedings of the Fifth International Machine Learning Conference*, 284-290.
- Tesauro G. 1995. Temporal Difference Learning and TD-Gammon, in *Communications of the ACM*, 38:58-68.