Orion Hess

Sources

Alpha-Beta pruning has been invented and reinvented a number of times throughout history, so pinning down a truly original source is difficult. It may have been John McCarthy at the 1956 Dartmouth Workshop on artificial intelligence.

That being said, T. A. Marsland, a Canadian computer scientist and game researcher has a paper that covers the topic
https://web.archive.org/web/20081030023047/http://www.cs.ualberta.ca/~tony/OldPapers/encyc.mac.pdf

This paper also covers trivial minimax.

Implementation Strategy

I will be implementing minimax with and without alpha-beta pruning for chess in a language called GDScript, a close derivative of Python. This allows me to easily integrate with the open source game engine Godot, making display and interaction with the program easy.

Depending on real world performance levels after implementation, I may offload some of the work to code in C#. Godot also supports this and C# is more performant due to its JIT compilation instead of GDScript's (Python's) interpretated style.

Initial Unit Tests

Minimax with and without alpha-beta pruning can be run on basic trees and does not need to actual be evaluating board states of chess. This makes testing significantly easier.

1. Check minimax on simple single depth tree
2. Check w/ ab pruning on simple single depth tree
3. Check that the visit order of nodes is as expected w/o ab pruning
4. Check that the visit order of nodes is as expected w/ ab pruning
5. Check that the correct path is chosen w/o ab pruning on basic random tree
6. Check that the correct path is chose w/ ab pruning on basic random tree
7. Check that specific nodes get properly pruned when using ab pruning
8. Check that specific nodes at different levels get pruned w/ ab pruning
9. Check worst case tree with ab pruning
10. Check best case tree with ab pruning