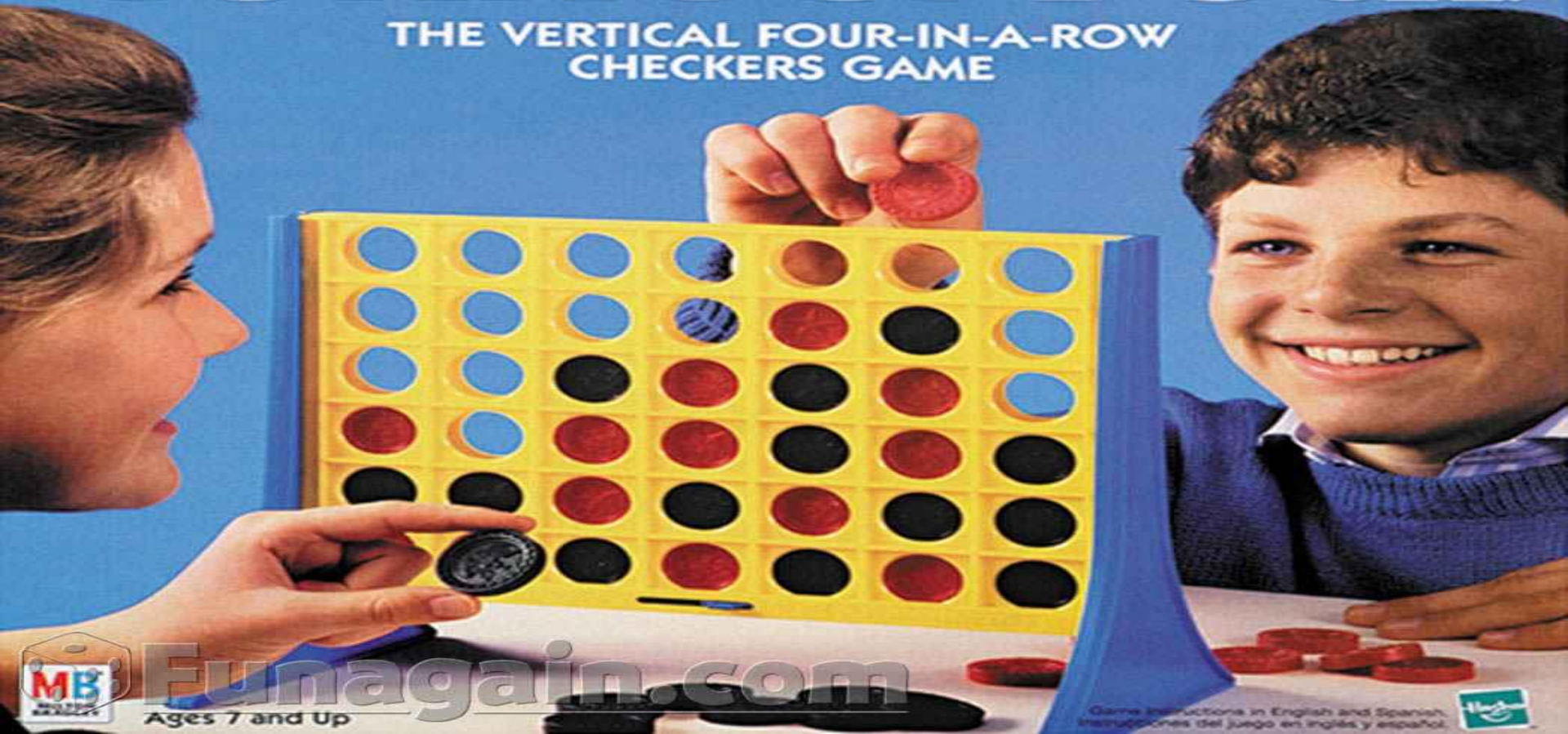
Solving Connect Four

By David I., David S., and Josh E.

Aka Team AlphaBeta

Connect Four®

THE VERTICAL FOUR-IN-A-ROW
CHECKERS GAME



Ages 7 and Up

Game instructions in English and Spanish.
Instrucciones del juego en inglés y español.

Connect Four

- We are making a Connect Four AI.
- A game state is a state of the board.
- Fully observable, deterministic, two-players, complete information.

High Level Theory

- We are using the adversarial MiniMax algorithm because it enables each agent to make the best move it can, while taking into account its opponent’s choices.
- We are using the Alpha-Beta pruning technique to eliminate subtrees representing bad or redundant moves.

Features

- Agents use “features” to analyze game states.
- “Features” are connected chains of like pieces that are unblocked on one or both sides (henceforth known as valid chains).
- Both agents want to assemble more, and longer, chains of their own color, and they want to break the opponent’s chains.

Feature Weights / Game State Value

- The int value of a game state for a given player is high if it’s good and low (negative) if it’s bad.
- Game state value is a function of which features exist on the board, multiplied by each feature’s weight.
- Opponent’s features have negative weights.

Our Approach

- Each turn, the turn player considers all possible states 8 (or 6, 4) moves ahead.
- Out of these possibilities, we create a game tree.
- MiniMax is performed on the tree to find the “best” move, by assuming the opponent will make optimal choices to maximize their score.
- We quantify the value of each state given user-entered features (weights) and general knowledge of the game.

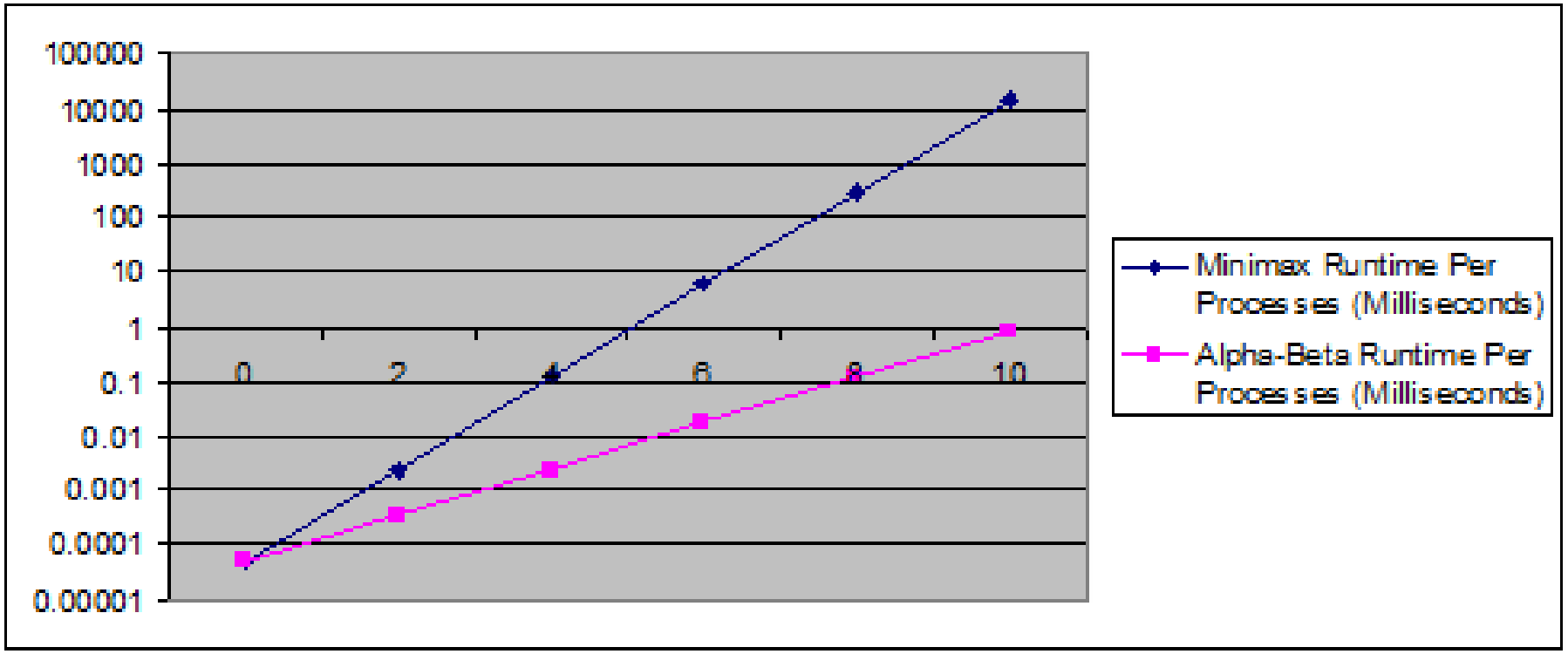
Our Approach

- Each player has a different set of weights for their own moves as well as their opponent’s moves.
- Thus, it’s as if each agent has a different idea of which moves to make given the state of the board.
- Our goal is to find an optimal set of weights that allows an agent to win or tie all its games.

Implementing Alpha-Beta

- Recursive, “bottom-up” algorithm.
- Because there are so many possible game states, (7^8), alpha-beta pruning eliminates TONS of redundant game states.
- Alpha beta pruning will improve runtime significantly

Runtime


100000
10000
1000
100
10
1
0.1
0.01
0.001
0.0001
0.00001
0
2
4
6
8
10
Minimax Runtime Per Processes (Milliseconds)
Alpha-Beta Runtime Per Processes (Milliseconds)

Measuring Success

- We are searching for a set of feature weights that results in the best gameplay we can achieve for an agent.
- We will test such a set against many other sets to determine if it is really the best.

Measuring Success

- Connect Four is solved, but we cannot compare our AI's gameplay to optimal gameplay because we don't have the computing power to optimize gameplay.
- We will test whether the moves chosen make sense, based on common sense and the skill we have achieved from playing hundreds of hours of Connect Four.

Our Code (So Far!)

- Feature weights and main method (Itkin).
- Feature finder, game state value calculator (Espinosa).
- Game tree creator (Su).
- Recursive Minimax function, returns the optimal move when given a game tree (Itkin).

Demonstration

Check it out!