Monte-Carlo Search Algorithms

Chang Liu
Andrew Tremblay
Outline

• Chess and Go
  o Large Game Trees

• Current Monte-Carlo Algorithms

• Current Codebases & Our Additions
  o Fuego
  o Gomba
Chess and Go

Chess
- Long established in AI
- Large Branching Factor,
  - Fixed board size (8x8)
- Games converge to win or loss

Go
- Used in AI More Recently
- Very Large Branching Factor
  - Varying Board Size (9x9 to 19x19)
- Games take much longer to converge
Game Trees

- Consist of States and Actions
- Picking an Action leads to a new State
- Find a winning state, choose the action leading to it
Large Game Trees

- Also consist of States and Actions
- How do you find the best action?
- How do you find one close to best?
UCT

• Upper Confidence Trees

• “Multi-Armed Bandit”
  o Simulated Regret

• Property of “Upper Confidence Bounds”
  o Balances Exploration Through Tree

• What this means
  o Always exploring the current best move
  o Converges towards optimal choice
UCT-RAVE

- Rapid Action Value Estimation (RAVE)
- Extension of basic UCT
- Updates the values across multiple states, rather than maintains the value on a per-action-state basis.
- AMAF (all moves as first) is a general name for this type of heuristic.
UCT-RAVE
Implementation - Fuego

- Pre-Existing Library of Go Algorithms
- UCT-RAVE already implemented
- Uses Go Text Protocol (GTP) to play against
  - Other Go Programs
    - GnuGo, MoGo
  - Humans
    - GoGui
- First program to defeat a 7 Dan Go player in 9x9 Go.
Fuego - Framework

- Large Codebase
- Only One External Library
- Multi-Threaded
- Open Source
- Extendable
Fuego - Experiments

Basic RAVE vs RAVE(Pool)

Time = 2T

Time = T + t
Disadvantages

Fuego et al. take a very long time to test.
   Play against separate algorithm for many games
   Change parameters
   Play again

Effectiveness of a new algorithm difficult to prove.
   Parameters difficult to optimize.

Cluster Jobs / Parallel Testing improves response of results

There is an alternative.
Artificial Game Tree

- Faster speed
- Better heuristic
- Easily twisted parameters (branching factors, depth, etc.)
Gomba

- Developed by Daniel Bjorge and John Schaeffer in 2010
- Mini-max Artificial Game Tree
- Lazy State Expansion
- Fast Action Simulation
- Fast Termination Evaluation
- Fast Heuristic Evaluation
- ... ...
Problem with Gomba

• AMAF algorithms - UCT RAVE algorithm
  o transposition table
  o global knowledge of the moves
• Structure of Gomba cannot support this type of algorithms
Our improvements

• Modified the tree structure to better support AMAF algorithms
• Add global knowledge of the moves
• Better correlation and consistency among the moves
• Modification to the lazy state expansion
Correlations among the moves
Adjust to the tree structure

Original lazy state expansion

New lazy state expansion
Improvements
Correlations

![Graphs showing search algorithm win rate with different correlation levels: No correlation, 50% correlation, 100% correlation.](image-url)
Conclusions & Future Work

• Improved Gomba testing framework
  ○ Better support of AMAF algorithms (UCT RAVE)

• Implemented RAVE-Pool in Fuego
  ○ With Variations
Köszönöm!

Advisors
• Levente Kocsis
• Gábor Sárközy
• Stanley Selkow

Daniel Bjorge & John Schaeffer
MTA-SZTAKI

All SZTAKI Colleagues
Kérdések?