CS4246 AI Planning and Decision Making

Monte Carlo Tree Search in Texas Holdem' Poker

Choo XianJun, Davin[1]
Lim Keng Kiat[1]
Wang Zi[1]

[1]Department of Computer Science
National University of Singapore

Monte Carlo Tree Search

- Why MCTS?

Monte Carlo Tree Search

- Why MCTS?
- Comparison against classic tree search algorithms
  - α-β pruning
  - A^* search

Overview

1. Monte Carlo
   - A statistical approach

Overview

1. Monte Carlo
   - A statistical approach
2. Tree Search
   - Search on a sequential problem domain

The 4 MCTS Phases

1. Selection
   - While we are at a visited node, *select* a child node
   - How? We shall discuss this later.

The 4 MCTS Phases

1. Selection

2. Expansion
   - If we reach an unvisited node, *expand*/create all possible child nodes
   - Mark node as visited and pick one of child nodes to explore

The 4 MCTS Phases

1. Selection
2. Expansion
3. Simulation
   - While we have not reached a terminal node, *simulate* a playthrough/rollout

The 4 MCTS Phases

1. Selection
2. Expansion
3. Simulation
4. Backpropagation
   - Compute reward at terminal node
   - *Backpropagate* reward back towards the root
   - Update relevant details needed to make selection decisions

The 4 MCTS Phases

1. Selection
2. Expansion
3. Simulation
4. Backpropagation

MCTS Outline

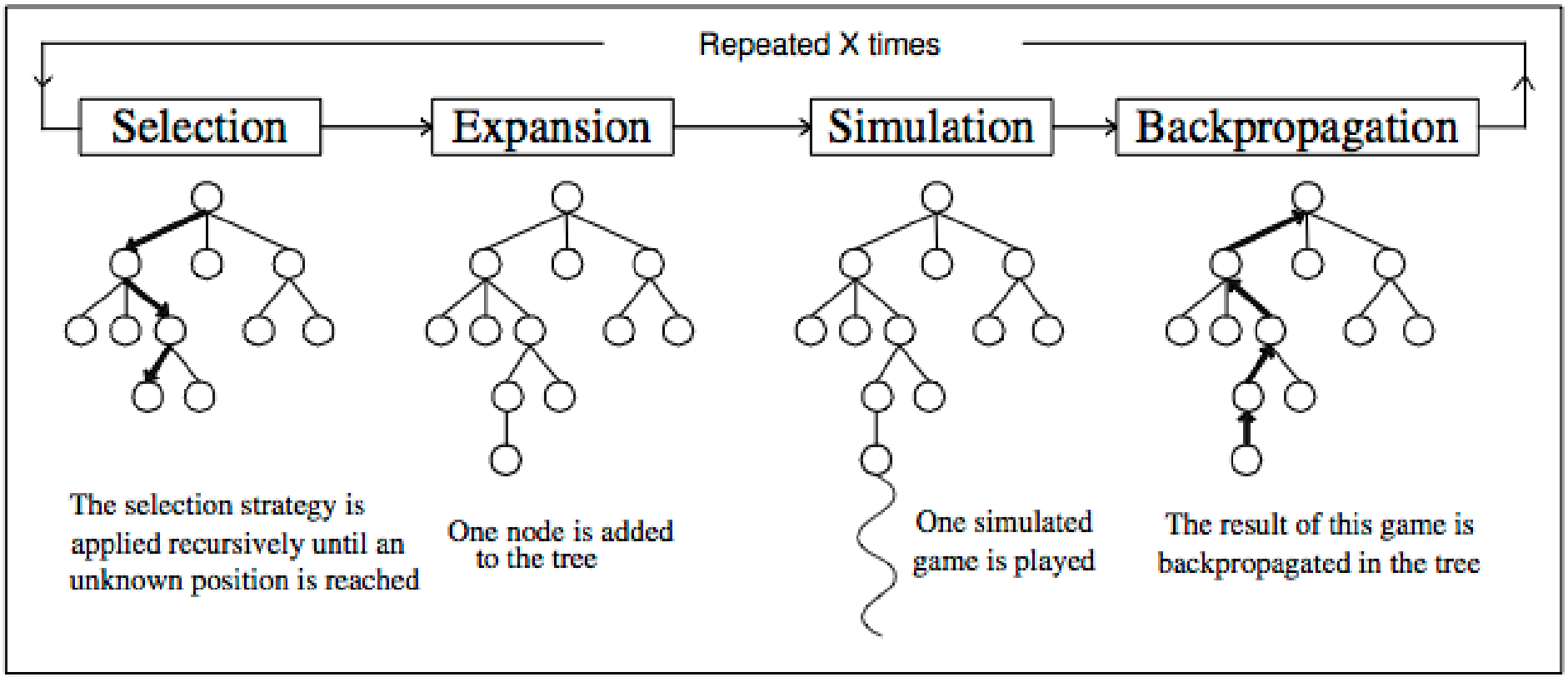

Figure : Monte Carlo Tree Search outline from Chaslot (2010)

Selection Choices

- Exploration vs. Exploitation

Selection Choices

- Exploration vs. Exploitation
- Upper Confidence Bound (UCB1)

$$UCB1 = \overline{X}_j + \sqrt{\frac{2 \ln n}{n_j}}$$

Selection Choices

- Exploration vs. Exploitation
- Upper Confidence Bound (UCB1)

$$UCB1 = \overline{X}_j + \sqrt{\frac{2 \ln n}{n_j}}$$

- Upper Confidence Bound for Trees (UCT)

$$UCT = \overline{X}_j + 2C_p\sqrt{\frac{2 \ln n}{n_j}}$$

Properties

- Ahueristic
  - Compared to other classical search algorithms that need intermediate evaluation functions/heuristics

Properties

- Ahueristic
  - Compared to other classical search algorithms that need intermediate evaluation functions/heuristics
- Anytime
  - Able to make sense even if stopped halfway during computation

Properties

- Ahueristic
  - Compared to other classical search algorithms that need intermediate evaluation functions/heuristics
- Anytime
  - Able to make sense even if stopped halfway during computation
- Asymmetric
  - Favour more promising nodes

Asymmetric Tree Growth

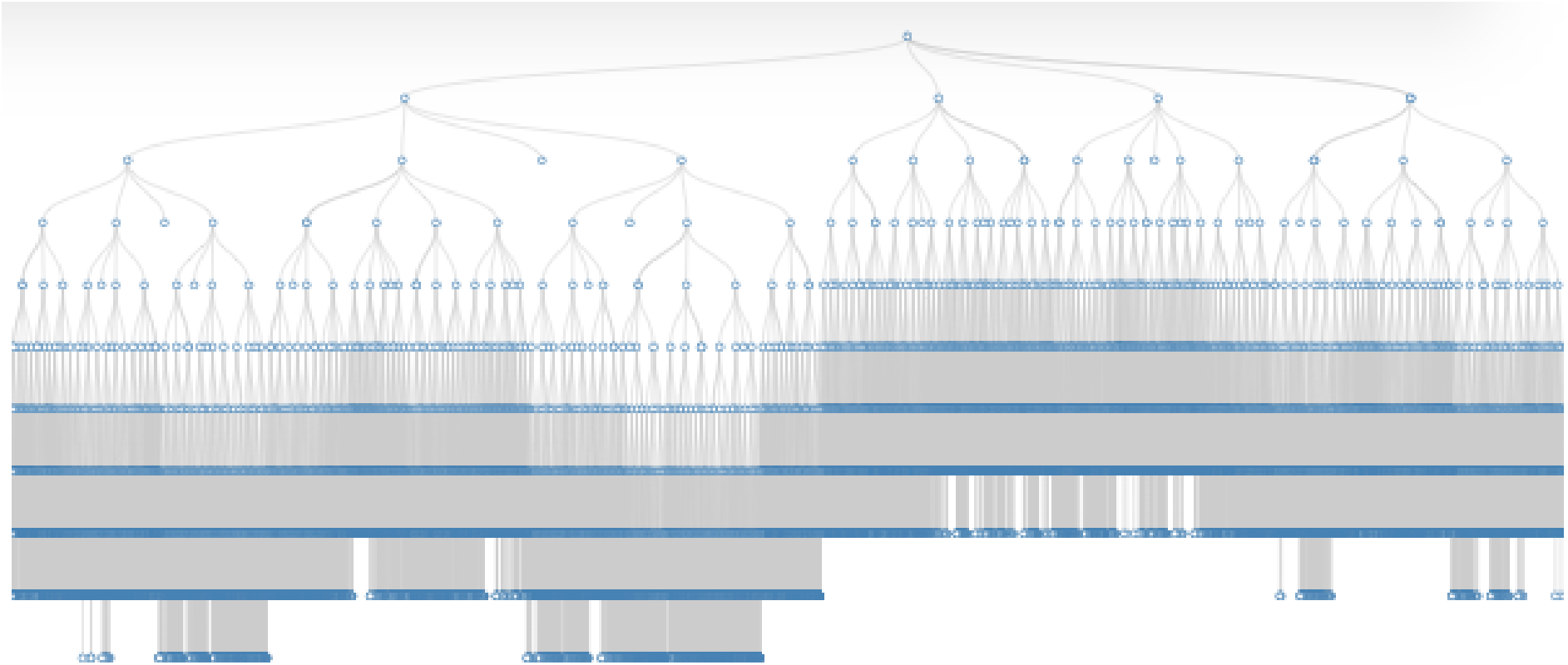

Figure : Illustration of asymmetric search tree of our MCTS Poker bot

Extensions to MCTS

- Parallelization
  - Leaf/Root/Tree parallelization

Extensions to MCTS

- Parallelization
  - Leaf/Root/Tree parallelization
- Re-using search trees
  - Start search tree for new node from appropriate branch of old node

Extensions to MCTS

- Parallelization
  - Leaf/Root/Tree parallelization
- Re-using search trees
  - Start search tree for new node from appropriate branch of old node
- Opponent Modelling
  - Vanilla MCTS assumes uniform distribution over opponent's actions

Extensions to MCTS

- Parallelization
  - Leaf/Root/Tree parallelization
- Re-using search trees
  - Start search tree for new node from appropriate branch of old node
- Opponent Modelling
  - Vanilla MCTS assumes uniform distribution over opponent's actions
- Adaptive Play
  - Able to detect change in opponent's strategy/playing style

2 Player Limit Texas Holdem' Poker

- Rules
  - Hand strength: Royal Flush $>$ Straight Flush $>$ Four of a Kind $>$ ...
  - Small blinds, Big blinds
  - Actions: Fold, Check, Call, Raise (Small), Raise (Big)
  - Stages: Deal, Flop, Turn, River

2 Player Limit Texas Holdem' Poker

- Rules
  - Hand strength: Royal Flush $>$ Straight Flush $>$ Four of a Kind $>$ ...
  - Small blinds, Big blinds
  - Actions: Fold, Check, Call, Raise (Small), Raise (Big)
  - Stages: Deal, Flop, Turn, River
- Properties
  - Partially observable
  - Non-deterministic
  - Large belief space $\rightarrow$ We approximate with MCTS

2 Player Limit Texas Holdem' Poker

- Rules
  - Hand strength: Royal Flush $>$ Straight Flush $>$ Four of a Kind $>$ ...
  - Small blinds, Big blinds
  - Actions: Fold, Check, Call, Raise (Small), Raise (Big)
  - Stages: Deal, Flop, Turn, River
- Properties
  - Partially observable
  - Non-deterministic
  - Large belief space $\rightarrow$ We approximate with MCTS
- Alternative approaches (Game theory, etc)

Implementation

- Python 2.7
- Demo

Where can we go from here?

- Look into MCTS extensions
- Extend our MCTS bot to Multiplayer No Limit Texas Holdem' Poker
- Hook up our MCTS bot with an actual Poker game client and see how well it fares

Thank you for your time

- For references and source codes, refer to report.
- Questions?