CMPUT 396 – Prologue

Puzzles: activities that can be solved by single agent search (sliding tile puzzles, Rubik's Cube, maze, etc.)

Game: activities that involve an element of competition, competing against something (another player, AI, time pressure, etc.) You might lose through the action of another agent. You can lose a game, you can't lose a puzzle.

3000 years ago, people starting playing Go. It seemed to originate in China, but it is popular in Japan and Korea as well. Go is the Japanese name, Weiqi is the Chinese name, and Baduk is the Korean name.

In 1985, Hayward became interested in the game of Hex.

By 2003, the status of computer players playing games were very different. In Hex, computers could beat some human players, but not the strongest. In Chess, bots had already reigned supreme over top humans (1996, won first game, 1997 won first match Deep Blue vs. Kasparov). In Go, every researcher on Go could beat their own bot.

Go is a lot more complex than something like Chess because the branching factor is much higher, and Chess has the easily-computed heuristic scoring function (1 point for a pawn, 3 for a knight/bishop, etc.) for arbitrary positions. In Go it's also really hard to see who will win at any point throughout the game.

Monte Carlo Tree Search (MCTS):

- Selection: start at the root and select the most promising leaf (the root is the current game state and the leaf is any node from which no simulation has been initiated yet).
- Expansion: unless the leaf node ends the game, create child nodes (valid moves from current game position) from this node and choose one
- Simulation: complete one random playout/rollout from the node selected (for example, choose uniform random moves until game is decided)
- Back Propagation: use the result from the playout to update information in the nodes on the path from the child node back to the root

MCTS asks:

- Which child should I go to next?
- How many times have I been to this node?
- What is the winning rate I have experienced from following this path?

2006 was the beginning of the Computer Go revolution.

- Remi Coulon & Crazy Stone: one of the first computer Go programs to utilize a modern variant of MCTS

    - Rated 5 dan in January 2012, rated 6 dan in March 2014
- Kocsis and Szepesvari (at DeepMind in London, on UAlberta faculty) & UCT (new algorithm)
    - UCT applies bandit ideas (to guide selective sampling of actions in rollout-based planning) to guide Monte-Carlo planning, combining MCTS and maximizing profit in slot machines
- Gelly, Wang, Munos and Teytund & Modifications of UCT
    - Developed a Monte-Carlo Go program, MoGo, which is the first computer Go program using UCT

Go ranking system:

- **Kyu** ranks are considered *student* ranks
    - Beginners who have just learned the game are usually around 30th kyu and as they progress their kyu ranking goes down (lower kyu ranking is better)
    - Best attainable kyu ranking is 1st kyu
- **Dan** ranks are considered *master* ranks
    - Dan ranking are when players progress past kyu
    - First ranking is 1st dan and moving upwards (higher dan ranking is better)
    - 1st dan is the equivalent of a black belt, but the very best players can continue in their professional dan rank

Rank Type	Range	Stage
Double-digit *kyu* (級,급) (*geup* in Korean)	30–20k	Beginner
Double-digit *kyu* (abbreviated: DDK)	19–10k	Casual player
Single-digit *kyu* (abbreviated: SDK)	9–1k	Intermediate amateur
Amateur *dan* (段,단)	1–7d	Advanced amateur
Professional *dan* (段,단)	1–9p	Professional Player

In 2015, the top Go programs Zen and Crazy Stone were about 8 dan (amateur). In 2014 Crazy Stone (with a 4 stone handicap on its opponent) beat Norimota Yoda by 2.5 points. It was still weaker than the top amateur players and weaker than any pro.

January 28th 2016: Mastering the game of Go with deep neural networks and tree search

- AlphaGo Nature Paper
- David Silver – lead researcher (UAlberta), Aja Huang – lead programmer (UAlberta), Demis Hassbis - CEO DeepMind
- Their program had beat a human professional player in a 5-game match (Fan Hui)
    - Professionals don't usually want to play computers because they either think they will win or know they will look silly if they lose
- In October 2015, DeepMind had convinced a top European player (3dan) to play 10 games on a 19x19 board against AlphaGo. They played 5 games with normal time limits (5-0 for AlphaGo) and 5 with the Blitz (fast) version (3-2 for AlphaGo)

It was announced that AlphaGo would play Lee Sedol in March 2016 in Seoul. Lee Sedol is like the Roger Federer of Go, in 2016 he was not the top player, but had been in the top 10 for the last 10-15 years.

DeepMind was a pure AI research group that was private (not affiliated with an institution). It was started by Demis Hassabis and friends. David Silver was brought on early (did his PhD at U of A, supervised by Rich Sutton and Martin Mueller). Aja Huang was the lead programmer (did his post doc here, supervised by Martin Mueller and Ryan Hayward).

There were 20 authors on the paper, which is interesting because usually there are something like 5 authors and 2 did all the work, but in this case it was 20 world experts employed by one institution.

AlphaGo vs. Lee Sedol match predictions:

- In the AlphaGo vs Fan Hui match, AlphaGo made many sub-optimal moves and threw away points. They believed that a top pro should easily beat that version of AlphaGo.
- However, some sub-optimal moves were not errors because AlphaGo picks moves that maximizes its probability of winning, not the margin by which it wins.
- Also, between the matches the AlphaGo algorithm changes and the AlphaGo neural networks trained nonstop and improved.

In March 2016, Alpha Go played Lee Sedol and won 4-1. In game 1, Lee Sedol tried unusual early moves to throw AlphaGo off its game, but it failed. In game 2, move 37 was to Hayward, when the era of human supremacy in Go ended. Move 37 was a shoulder hit in the middle (usually you make a move near the edge). This was a very unusual move for a human to make, but in some ways redefines some things about Go that AlphaGo was choosing creative moves like this.

Search: exploring possible moves/actions. We can solve puzzles by brute force if the search space is small enough, but usually we need more guiding information to search in a smarter way.