CMPUT 396: Solving Go

Important: early win detection, that is determining the final score before reaching a terminal position and recognizing blocks that cannot be killed.

Solving Go is much more complicated than solving Hex.

John Tromp wrote code to solve 2x2 Go. In theory, with enough time and memory, any Go state can be solved by minimax. But in practice, this is infeasible for boards that are 8x8 or bigger.

A few key things:

- Move ordering: make the important moves first (check if there is a pass first), this gives you a much smaller tree to search.
- Simplest possible, no suicide, TT rules, positional superko

2 reasons solving Go is hard:

1) Capture: stones can be removed but that position can be refilled later with another colour
2) Ko: you have to remember the history to know if that move is allowed

Ko: means “eternity” in Japanese, prohibit recurrences of an earlier board position (which could lead the game into an inescapable loop of players trading moves).

Positional superko: can’t repeat a position that has been played before (regardless of whose turn it was when that position was first reached).

Situational superko: prevents the repetition of the same situation (forbids a player from reaching a position that that player reached previously with one of their moves, the other player can play it though).

Solving 1xn Go for n over 12 is an open problem.

Many early-win-detection observations do no apply in linear Go. Linear Go blocks have no interiour region, so blocks have no true eyes. There are no local safe linear Go patterns. We need linear Go early-win-detection and pruning observations.

Hayward had a student solving 1x6 board with positional superko and no suicide.

- Positional superko: makes search space a bit smaller because some moves are forbidden
- No suicide: this is commonly the rule when people play Go and again, it makes the search space smaller

Using these two rules reduces the number of legal moves that must be looked at.

Solving 2x2 Go

Minimax score for 2x2 Go is black wins by (at least) 1 (proof tree in handout).

How can we show that black wins by at most 1 (and so black wins by exactly 1)?
We could show white's strategy too. If under minimax white can play to lose by at most 1, then black wins by exactly 1 under minimax.