4 Heuristic Search

4.0 Introduction
4.1 An Algorithm for Heuristic Search
4.2 Admissibility, Monotonicity, and Informedness

4.3 Using Heuristics in Games
4.4 Complexity Issues
4.5 Epilogue and References
4.6 Exercises

Additional references for the slides:
Robert Wilensky’s CS188 slides:
www.cs.berkeley.edu/%7wilensky/cs188/lectures/index.html
Tim Huang’s slides for the game of Go.
A variant of the game nim

- A number of tokens are placed on a table between the two opponents
- A move consists of dividing a pile of tokens into two nonempty piles of different sizes
- For example, 6 tokens can be divided into piles of 5 and 1 or 4 and 2, but not 3 and 3
- The first player who can no longer make a move loses the game
- For a reasonable number of tokens, the state space can be exhaustively searched
State space for a variant of nim

```
  7
 /   \
6-1   5-2   4-3
 /   /   / \
5-1-1 4-2-1 3-2-2 3-3-1
 /   /   /   / \
4-1-1-1 3-2-1-1 2-2-2-1
 /   /   / \
3-1-1-1-1 2-2-1-1
 /   \
2-1-1-1-1-1
```
Exhaustive minimax for the game of nim
Two people games

• One of the earliest AI applications

• Several programs that compete with the best human players:
  • Checkers: beat the human world champion
  • Chess: beat the human world champion (in 2002 & 2003)
  • Backgammon: at the level of the top handful of humans
  • Go: no competitive programs
  • Othello: good programs
  • Hex: good programs
Search techniques for 2-person games

• The search tree is slightly different: It is a *two-ply tree* where levels alternate between players.

• Canonically, the first level is “us” or the player whom we want to win.

• Each final position is assigned a payoff:
  - win (say, 1)
  - lose (say, -1)
  - draw (say, 0)

• We would like to maximize the payoff for the first player, hence the names MAX & MINIMAX.
The search algorithm

- The root of the tree is the current board position, it is MAX’s turn to play.

- MAX generates the tree as much as it can, and picks the best move assuming that Min will also choose the moves for herself.

- This is the *Minimax algorithm* which was invented by Von Neumann and Morgenstern in 1944, as part of game theory.

- The same problem with other search trees: the tree grows very quickly, exhaustive search is usually impossible.
Special technique 1

• MAX generates the full search tree (up to the leaves or terminal nodes or final game positions) and chooses the best one: win or tie

• To choose the best move, values are propagated upward from the leaves:
  - MAX chooses the maximum
  - MIN chooses the minimum

• This assumes that the full tree is not prohibitively big

• It also assumes that the final positions are easily identifiable

• We can make these assumptions for now, so let’s look at an example
Two-ply minimax applied to X’s move near the end of the game (Nilsson, 1971)
Special technique 2

• Notice that the tree was not generated to full depth in the previous example

• When time or space is tight, we can’t search exhaustively so we need to implement a cut-off point and simply not expand the tree below the nodes who are at the cut-off level.

• But now the leaf nodes are not final positions but we still need to evaluate them: use heuristics

• We can use a variant of the “most wins” heuristic
Heuristic measuring conflict

X has 6 possible win paths: X
O has 5 possible wins: O

E(n) = 6 − 5 = 1

X has 4 possible win paths; X
O has 6 possible wins: O

E(n) = 4 − 6 = −2

X has 5 possible win paths; X
O has 4 possible wins: O

E(n) = 5 − 4 = 1
**Calculation of the heuristic**

- \( E(n) = M(n) – O(n) \)
  - \( M(n) \) is the total of My (MAX) possible winning lines
  - \( O(n) \) is the total of Opponent’s (MIN) possible winning lines
  - \( E(n) \) is the total evaluation for state \( n \)

- Take another look at the previous example

- Also look at the next two examples which use a cut-off level (a.k.a. *search horizon*) of 2 levels
Two-ply minimax applied to the opening move of tic-tac-toe (Nilsson, 1971)
Two-ply minimax and one of two possible second MAX moves (Nilsson, 1971)
Minimax applied to a hypothetical state space (Fig. 4.15)
Special technique 3

• Use alpha-beta pruning

• Basic idea: if a portion of the tree is obviously good (bad) don’t explore further to see how terrific (awful) it is

• Remember that the values are propagated upward. Highest value is selected at MAX’s level, lowest value is selected at MIN’s level

• Call the values at MAX levels $\alpha$ values, and the values at MIN levels $\beta$ values
The rules

• Search can be stopped below any MIN node having a beta value less than or equal to the alpha value of any of its MAX ancestors

• Search can be stopped below any MAX node having an alpha value greater than or equal to the beta value of any of its MIN node ancestors
Example with MAX

As soon as the node with value 2 is generated, we know that the beta value will be less than 3, we don’t need to generate these nodes (and the subtree below them)
As soon as the node with value 6 is generated, we know that the alpha value will be larger than 6, we don’t need to generate these nodes (and the subtree below them).
Alpha-beta pruning applied to the state space of Fig. 4.15

A has $\beta = 3$ (A will be no larger than 3)
B is $\beta$ pruned, since $5 > 3$
C has $\alpha = 3$ (C will be no smaller than 3)
D is $\alpha$ pruned, since $0 < 3$
E is $\alpha$ pruned, since $2 < 3$
C is 3
Number of nodes generated as a function of branching factor $B$, and solution length $L$ (Nilsson, 1980)
Informal plot of cost of searching and cost of computing heuristic evaluation against heuristic informedness (Nilsson, 1980)
Othello (a.k.a. reversi)

- 8x8 board of cells
- The tokens have two sides: one black, one white
- One player is putting the white side and the other player is putting the black side
- The game starts like this:
Othello

- The game proceeds by each side putting a piece of his own color
- The winner is the one who gets more pieces of his color at the end of the game
- Below, white wins by 28
Othello

- When a black token is put onto the board, and on the same horizontal, vertical, or diagonal line there is another black piece such that every piece between the two black tokens is white, then all the white pieces are flipped to black.

- Below there are 17 possible moves for white.
Othello

• A move can only be made if it causes flipping of pieces. A player can pass a move iff there is no move that causes flipping. The game ends when neither player can make a move

• the snapshots are from www.mathewdoucette.com/artificialintelligence

• the description is from home.kkto.org:9673/courses/ai-xhtml

• AAAI has a nice repository: www.aaai.org
Click on AI topics, then select “games & puzzles” from the menu
Hex

- Hexagonal cells are arranged as below. Common sizes are 10x10, 11x11, 14x14, 19x19.
- The game has two players: Black and White
- Black always starts (there is also a swapping rule)
- Players take turns placing their pieces on the board
Hex

- The object of the game is to make an uninterrupted connection of your pieces from one end of your board to the other

- Other properties
  - First player always wins
  - No ties
Hex

- Invented independently by Piet Hein in 1942 and John Nash in 1948.
- Every empty cell is a legal move, thus the game tree is wide $b = \sim 80$ (chess $b = \sim 35$, go $b = \sim 250$)
- Determining the winner (assuming perfect play) in an arbitrary Hex position is PSPACE-complete [Rei81].
- How to get knowledge about the “potential” of a given position without massive game-tree search?
Hex

- There are good programs that play with heuristics to evaluate game configurations
- hex.retes.hu/six
- home.earthlink.net/~vanshel
- cs.ualberta.ca/~javhar/hex
- www.playsite.com/t/games/board/hex/rules.html
The Game of Go

Go is a two-player game played using black and white stones on a board with 19x19, 13x13, or 9x9 intersections.
The Game of Go

Players take turns placing stones onto the intersections. Goal: surround the most territory (empty intersections).
The Game of Go

Once placed onto the board, stones are not moved.
The Game of Go
The Game of Go
The Game of Go
The Game of Go
The Game of Go
The Game of Go

A **block** is a set of adjacent stones (up, down, left, right) of the same color.
The Game of Go

A **block** is a set of adjacent stones (up, down, left, right) of the same color.
The Game of Go

A liberty of a block is an empty intersection adjacent to one of its stones.
The Game of Go
The Game of Go
The Game of Go

If a block runs out of liberties, it is captured. Captured blocks are removed from the board.
The Game of Go

If a block runs out of liberties, it is captured. Captured blocks are removed from the board.
The Game of Go

If a block runs out of liberties, it is captured. Captured blocks are removed from the board.
The Game of Go

The game ends when neither player wishes to add more stones to the board.
The Game of Go

The player with the most enclosed territory wins the game. (With *komi*, White wins this game by 7.5 pts.)
Alive and Dead Blocks

White can capture by playing at A or B. Black can capture by playing at C. Black can’t play at D and E simultaneously.

With only one eye, these stones are dead. No need for Black to play at C.

With two eyes at D and E, these White stones are alive.
Example on 13x13 Board

What territory belongs to White? To Black?
Example on 13x13 Board

Black ahead by 1 point. With *komi*, White wins by 4.5 pts.
Challenges for Computer Go

Much higher search requirements

- Minimax game tree has $O(b^d)$ positions
- In chess, $b = \sim 35$ and $d = \sim 100$ half-moves
- In Go, $b = \sim 250$ and $d = \sim 200$ half-moves
- However, 9x9 Go seems almost as hard as 19x19

Accurate evaluation functions are difficult to build and computationally expensive

- In chess, material difference alone works fairly well
- In Go, only 1 piece type with no easily extracted features

Determining the winner from an arbitrary position is PSPACE-hard (Lichtenstein and Sipser, 1980)
State of the Art

Many Faces of Go v.11 (Fotland), Go4++ (Reiss), Handtalk/Goemate (Chen), GNUGo (many), etc.

Each consists of a carefully crafted combination of pattern matchers, expert rules, and selective search

Playing style of current programs:

- Focus on safe territories and large frameworks
- Avoid complicated fighting situations

Rank is about 6 kyu, though actual playing strength varies from opening (stronger) to middle game (much weaker) to endgame (stronger)