1. Introduction

Jonathan Schaeffer
jonathan@cs.ualberta.ca
www.cs.ualberta.ca/~jonathan

AI and Search

- Search is an integral part of many artificial intelligence applications
- It takes many forms:
  - Explicit (e.g. game-playing programs, optimization, path-finding, planning, web agents, theorem proving, satisfiability, belief networks, NP-complete problems, DNA sequence alignment)
  - Implicit (e.g. Prolog deductions)
- Search is the heart of artificial intelligence

Search and Knowledge

- Some will argue that AI consists of only two fundamental concepts:
- Knowledge
  - Representation and manipulation of factual and probabilistic knowledge
- Search
  - Explore knowledge alternatives to arrive at the best quality answer

Search is...

- Search is knowledge!
- Enhance the quality of explicit knowledge
  - Refine the values of heuristic knowledge
- Reduce knowledge required
  - No need to include explicit knowledge that will be uncovered by the search
  - Eliminate the knowledge acquisition bottleneck
- Used to uncover implicit knowledge
  - Chess with a random number evaluation function
  - Bridge/poker/scrabble using simulations

AI Textbooks

- Open any introductory book on AI and examine the section on search
  - Alpha-beta algorithm (games are popular)
  - A* (puzzles, planning and optimization)
  - AO* (theorem proving)
  - Simulations (maybe, for probabilistic domains)
  - SAT (maybe, for satisfiability)

Changing Perceptions

- Over the years, the percent of pages devoted to search in introductory AI books has decreased
  - Neilson (1980) 17%
  - Rich (1983) 31%
  - Russell and Norvig (1995) 12%
  - Poole, Mackworth and Goebel (1998) 10%
  - Russell and Norvig (2003) 13%

Why?

- Search algorithms are “well understood”
- No major advances in our basic understanding of the algorithms
- But… is the algorithm alone enough to tell you how to use search in a high-performance program?

Search Reality

- Most of the core AI search algorithms can be expressed in 20 lines of code
- Is search really that simple?

“If the search algorithm is really 20 lines of code, then why is my search routine over 20 pages of code?”

M. Newborn, 1985

Search is all about...

- Not the search algorithm!
  - Usually trivial decision made based on the application to be solved
- The search enhancements!
  - Standard algorithms are often impractical
  - The enhancements can reduce the execution time of a search by orders of magnitude

The Message Is...

- Efficient search is all about search enhancements
- Given an application domain, choice of algorithm is usually trivial
- 99% of the effort is spent implementing, debugging, tuning and analyzing search enhancements
- The AI textbooks have it backwards

Brute-force Search

- Search enhancements make it possible for "brute-force" techniques to solve a problem
- Paradoxical that a "dumb" and exhaustive searcher can out-perform a "smart" and selective searcher
- This is one of the major advances in AI
  - It took many years to get grudging respect from the AI community

The Challenge!

- Build high-performance search programs
- Start with the basic algorithm and improve the search through:
  - Better quality evaluation function
  - Combination of search enhancements

Heuristic Search

- Combination of search and knowledge to explore a state space to find the best quality answer
- The goal is to dampen or eliminate the exponential growth of the search tree

Search Goals

- Optimizing
  - Optimal solution(s)
  - May not be achievable in reasonable time
- Satisficing
  - Come up with the best quality decision within given resource constraints
  - Solution quality can be significantly worse than an optimal solution

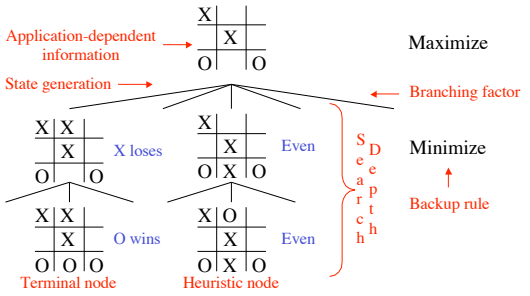
Search Knowledge

- Optimal
  - *Terminal nodes* of search tree are well-defined by the application domain
  - E.g., win/loss/draw; length/cost of solution
- Satisficing
  - Can't reach all the terminal nodes
  - Stop search at *leaf nodes* and approximate result
  - Use a *heuristic evaluation function*

Ingredients in a Searcher?

- Application-dependent information
- Successor state generator
- Algorithm
  - Next node to expand
  - Backup rule
- Heuristic evaluation function

Example



Algorithms?

- Adversary search
  - Alpha-beta and variants
- Single-agent search
  - A* and variants
- Imperfect information
  - Simulations
- Stochastic
  - Expecticmax and variants
- Other
  - Many other algorithms for handling specific cases

It's All About the Algorithm?

- Choice of algorithm is often trivial given a statement of the problem
  - Pre-packaged code is usually readily available
  - Just need to put in application-dependent information

It's All About Enhancements!

- Generic search algorithms get poor performance
- There are a plethora of enhancements in the literature that can dramatically improve the efficiency of search

Exponential Search Growth!?

- Branching factor = *b*
- Search depth = *d*
- Naïve search = b^d nodes
- Smart search?

IDA* Example

15-Puzzle (test set of 100 positions):

Full-width search	1,000,000,000,000,000,000,000,000,000,000
DAG, not a tree	~10,000,000,000,000
IDA*	36,302,808,031
Enhanced search [1]	21,261,747
Further work [2]	~1,000,000
Hot off the press! [3]	19,540

Alpha-beta Example

Checkers program:

Minimax search:	100,000,000,000,000
Alpha-beta search:	310,000,000
Enhanced search:	7,000,000

Conclusion

Need I say more?

References

[1] Joseph Culberson and Jonathan Schaeffer. “Pattern Databases”, *Computational Intelligence*, Vol. 14, No. 4, pp. 318-334, 1998.

[2] Richard Korf and Ariel Felner, “Disjoint Pattern Database Heuristics”, Artificial Intelligence, vol. 134, no. 1-2, pp. 9-22, 2002.

[3] Ariel Felner, Robert Holte, Jonathan Schaeffer, and Uzi Zahavi. Unpublished, 2004.