Problem Spaces & Search

CSE 473

473 Topics

Agents & Environments
Problem Spaces
Search & Constraint Satisfaction
Knowledge Repr’n & Logical Reasoning
Machine Learning
Uncertainty: Repr’n & Reasoning
Dynamic Bayesian Networks
Markov Decision Processes

Weak Methods

“In the knowledge lies the power...”
[Feigenbaum]

But what if no knowledge???

Generate & Test

As weak as it gets...

Works on semi-decidable problems!

Example:
Fragment of 8-Puzzle Problem Space

Search thru a Problem Space / State Space

Input:
- Set of states
- Operators [and costs]
- Start state
- Goal state [test]

Output:
- Path: start ⇒ a state satisfying goal test
- [May require shortest path]
**Example: Route Planning**

Input:
- Set of states
- Operators [and costs]
- Start state
- Goal state (test)

Output:

**Example: N Queens**

Input:
- Set of states
- Operators [and costs]
- Start state
- Goal state (test)

Output

**Example: Blocks World**

Input:
- Set of states
- Operators [and costs]
- Start state
- Goal state (test)

A plan which provably achieves

The desired world configuration

**Multiple Problem Spaces**

Real World
- States of the world (e.g. block configurations)
- Actions (take one world-state to another)

Robot's Head

Problem Space 1
- PS states = models of world states
- Operators = plan modification ops

Problem Space 2
- PS states = partially spec. plan
- Operators = plan modification ops

**Classifying Search**

**GUESSING (“Tree Search”)**
- Guess how to extend a partial solution to a problem.
- Generates a tree of (partial) solutions.
- The leaves of the tree are either “failures” or represent complete solutions.

**SIMPLIFYING (“Inference”)**
- Infer new, stronger constraints by combining one or more constraints (without any “guessing”).
  
  Example: $X + 2Y = 3$
  $X + Y = 1$
  Therefore $Y = 2$

**WANDERING (“Markov chain”)**
- Perform a (biased) random walk through the space of (partial or total) solutions.

---

**Guessing - State Space Search**
1. BFS
2. DFS
3. Iterative Deepening
4. Bidirectional
5. Best-first search
6. A*
7. Game tree
8. Davis-Putnam (logic)
9. Cutset conditioning (probability)

**Simplification - Constraint Propagation**
1. Forward Checking
2. Path Consistency (Waltz labeling, temporal algebra)
3. Resolution
4. “Bucket Algorithm”

**Wandering - Randomized Search**
1. Hillclimbing
2. Simulated annealing
3. Walksat
4. Monte-Carlo Methods
### Search Strategies v2

**Blind Search**
- Depth first search
- Breadth first search
- Iterative deepening search
- Iterative broadening search

**Informed Search**
- Constraint Satisfaction
- Adversary Search

---

### Depth First Search

Maintain stack of nodes to visit

**Evaluation**
- **Complete?**
  - Not for infinite spaces
- **Time Complexity?**
  - $O(b^d)$
- **Space Complexity?**
  - $O(d)$

---

### Breadth First Search

Maintain queue of nodes to visit

**Evaluation**
- **Complete?**
  - Yes
- **Time Complexity?**
  - $O(b^d)$
- **Space Complexity?**
  - $O(b^d)$

---

### Iterative Deepening Search

DFS with limit; incrementally grow limit

**Evaluation**
- **Complete?**
  - Yes
- **Time Complexity?**
  - $O(b^d)$
- **Space Complexity?**
  - $O(d)$

---

### Cost of Iterative Deepening

<table>
<thead>
<tr>
<th>b</th>
<th>ratio ID to DFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>1.5</td>
</tr>
<tr>
<td>10</td>
<td>1.2</td>
</tr>
<tr>
<td>25</td>
<td>1.06</td>
</tr>
<tr>
<td>100</td>
<td>1.02</td>
</tr>
</tbody>
</table>
When to Use Iterative Deepening

N Queens?

Search Space with Uniform Structure

Search Space with Clustered Structure

Iterative Broadening Search

What if know solutions lay at depth N?  
No sense in doing iterative deepening

Forwards vs. Backwards

vs. Bidirectional
**Problem**

All these methods are slow (blind)

Solution → add guidance ("heuristic estimate") → "informed search"

... next time ...

**Recap: Search thru a Problem Space / State Space**

Input:
- Set of states
- Operators [and costs]
- Start state
- Goal state [test]

Output:
- Path: start ⇒ a state satisfying goal test
  [May require shortest path]

**Cryptarithmetic**

Input:
- Set of states
- Operators [and costs]
- Start state
- Goal state (test)

Output:

**Concept Learning**

Labeled Training Examples

- <p1, blond, 32, mc, ok>
- <p2, red, 47, visa, ok>
- <p3, blond, 23, cash, ter>
- <p4, ...

Output: f: <pn...> ⇒ {ok, ter}

**Symbolic Integration**

E.g. \[ \int x^2 e^x \, dx = e^x(x^2-2x+2) + C \]

Operators:
- Integration by parts
- Integration by substitution
  ...

© Daniel S. Weld