Lecture 5 of 42

Informed Search: Best First Search (Greedy, A/A*) and Heuristics
Discussion: Project Topics 5 of 5

Friday, 04 September 2009

William H. Hsu
Department of Computing and Information Sciences, KSU

KSOL course page: http://snipurl.com/v9v3
Course web site: http://www.kddresearch.org/Courses/Fall-2009/CIS730
Instructor home page: http://www.cis.ksu.edu/~bhsu

Reading for Next Class:
Section 4.3, p. 110 - 118, Russell & Norvig 2nd edition
Instructions for writing project plans, submitting homework

Lecture Outline

• Reading for Next Class: Section 4.3, R&N 2nd edition
• Coming Week: Chapter 4 concluded, Chapter 5
  • Properties of search algorithms, heuristics
  • Local search (hill-climbing, Beam) vs. nonlocal search
  • Genetic and evolutionary computation (GEC)
  • State space search: graph vs. constraint representations
• Today: Sections 4.1 (Informed Search), 4.2 (Heuristics)
  • Properties of heuristics: consistency, admissibility, monotonicity
  • Impact on A/A*
• Next Class: Section 4.3 on Local Search and Optimization
  • Problems in heuristic search: plateaux, “foothills”, ridges
  • Escaping from local optima
  • Wide world of global optimization: genetic algorithms, simulated annealing
• Next Week: Chapter 5 on CSP
Search-Based Problem Solving: Quick Review

- **function** `General-Search(problem, strategy)` returns a solution or failure
  - Queue: represents search frontier (see: Nilsson – OPEN / CLOSED lists)
  - Variants: based on “add resulting nodes to search tree”

Previous Topics

- Formulating problem
- Uninformed search
  - No heuristics: only \( g(n) \), if any cost function used
  - Variants: BFS (uniform-cost, bidirectional), DFS (depth-limited, ID-DFS)
- Heuristic search
  - Based on \( h \) – (heuristic) function, returns estimate of min cost to goal
  - \( h \) only: greedy (aka myopic) informed search
  - \( A^*/A^* \): \( f(n) = g(n) + h(n) \) – frontier based on estimated + accumulated cost

Today: More Heuristic Search Algorithms

- \( A^* \) extensions: iterative deepening (IDA\(^*\)), simplified memory-bounded (SMA\(^*\))
- Iterative improvement: hill-climbing, MCMC (simulated annealing)
- Problems and solutions (macros and global optimization)

Graph Search Example: Romanian Map Revisited

Greedy Search [1]: A Best-First Algorithm

- **function** Greedy-Search (problem) returns solution or failure
  - // recall: solution Option
  - return Best-First-Search (problem, h)

- Example of Straight-Line Distance (SLD) Heuristic: Figure 4.2 R&N
  - Can only calculate if city locations (coordinates) are known
  - Discussion: Why is $h_{SLD}$ useful?
    - Underestimate
    - Close estimate

- Example: Figure 4.3 R&N
  - Is solution optimal?
  - Why or why not?

---

Greedy Search [2]: Example

Adapted from slides © 2003 S. Russell & P. Norvig. Reused with permission.
Greedy Search [3]: Properties

- Similar to DFS
  - Prefers single path to goal
  - Backtracks
- Same Drawbacks as DFS?
  - Not optimal
    - First solution
    - Not necessarily best
    - Discussion: How is this problem mitigated by quality of $h$?
  - Not complete: doesn’t consider cumulative cost “so-far” ($g$)
- Worst-Case Time Complexity: $O(b^m)$ – Why?
- Worst-Case Space Complexity: $O(b^m)$ – Why?

Greedy Search [4]: More Properties

- Good Heuristics Reduce Practical Space/Time Complexity
  - “Your mileage may vary”: actual reduction
    - Domain-specific
    - Depends on quality of $h$ (what quality $h$ can we achieve?)
  - “You get what you pay for”: computational costs or knowledge required
- Discussions and Questions to Think About
  - How much is search reduced using straight-line distance heuristic?
  - When do we prefer analytical vs. search-based solutions?
  - What is the complexity of an exact solution?
  - Can “meta-heuristics” be derived that meet our desiderata?
    - Underestimate
    - Close estimate
  - When is it feasible to develop parametric heuristics automatically?
    - Finding underestimates
    - Discovering close estimates
Algorithm A/A* [1]: Methodology

- **Idea:** Combine Evaluation Functions $g$ and $h$
  - Get “best of both worlds”
  - **Discussion:** Importance of taking both components into account?
- **Function A-Search (problem)** returns solution or failure
  - // recall: solution Option
  - return Best-First-Search (problem, $g + h$)
- **Requirement:** Monotone Restriction on $f$
  - Recall: monotonicity of $h$
  - Requirement for completeness of uniform-cost search
  - Generalize to $f = g + h$
  - *aka* triangle inequality
- **Requirement for A = A*:** Admissibility of $h$
  - $h$ must be underestimate of true optimal cost ($\forall n . h(n) \leq h^*(n)$)

Algorithm A/A* [2]: Example

Path found: (Arad → Sibiu → Rimnicu Vilcea → Pitesti → Bucharest) _116_

Nodes found/scheduled (opened): {A, S, T, Z, F, O, RV, S, B, C, P}

Nodes visited (closed): {A, S, F, RV, P, B}
Algorithm A/A*[3]: Properties

- Completeness (p. 100 R&N 2e)
  - Expand lowest-cost node on fringe
  - Requires Insert function to insert into increasing order
- Optimality (p. 99-101 R&N 2e)
- Optimal Efficiency (p. 97-99 R&N 2e)
  - For any given heuristic function
  - No other optimal algorithm is guaranteed to expand fewer nodes
  - Proof sketch: by contradiction (on what partial correctness condition?)
- Worst-Case Time Complexity (p. 100-101 R&N 2e)
  - Still exponential in solution length
  - Practical consideration: optimally efficient for any given heuristic function

Algorithm A/A*[4]: Performance

- Admissibility: Requirement for A* Search to Find Min-Cost Solution
- Related Property: Monotone Restriction on Heuristics
  - For all nodes m, n such that m is a descendant of n: h(m) ≥ h(n) - c(n, m)
  - Change in h is less than true cost
  - Intuitive idea: “No node looks artificially distant from a goal”
- Discussion questions
  - Admissibility ⇒ monotonicity? Monotonicity ⇒ admissibility?
  - Always realistic, i.e., can always be expected in real-world situations?
  - What happens if monotone restriction is violated? (Can we fix it?)
- Optimality and Completeness
  - Necessarily and sufficient condition (NASC): admissibility of h
  - Proof: p. 99-100 R&N (contradiction from inequalities)
- Behavior of A*: Optimal Efficiency
- Empirical Performance
  - Depends very much on how tight h is
  - How weak is admissibility as a practical requirement?
Properties of Algorithm A/A*: Review

- Admissibility: Requirement for A* Search to Find Min-Cost Solution
- Related Property: Monotone Restriction on Heuristics
  - For all nodes m, n such that m is a descendant of n: h(m) ≥ h(n) - c(n, m)
  - Discussion questions
    - Admissibility ⇒ monotonicity? Monotonicity ⇒ admissibility?
    - What happens if monotone restriction is violated? (Can we fix it?)
- Optimality Proof for Admissible Heuristics
  - Theorem: If ∀n. h(n) ≤ h*(n), A* will never return a suboptimal goal node.
  - Proof
    - Suppose A* returns x such that ∃s. g(s) < g(x)
    - Let path from root to s be < n₀, n₁, ..., nᵢ > where nᵢ ≡ s
    - Suppose A* expands a subpath < n₀, n₁, ..., nᵢ > of this path
    - Lemma: by induction on j, s = nᵢ is expanded as well
      Base case: n₀ (root) always expanded
      Induction step: h(nᵢ−1) ≤ h*(nᵢ−1), so f(nᵢ−1) ≤ f(x), Q.E.D.
    - Contradiction: if s were expanded, A* would have selected s, not x

A/A*: Extensions (IDA*, RBFS, SMA*)

- Memory-Bounded Search (p. 101 – 104, R&N 2°)
  - Rationale
    - Some problems intrinsically difficult (intractable, exponentially complex)
    - “Something’s got to give” – size, time or memory? (“Usually memory”)
- Recursive Best–First Search (p. 101 – 102 R&N 2°)
- Iterative Deepening A* – Pearl, Korf (p. 101, R&N 2°)
  - Idea: use iterative deepening DFS with sort on f – expands node iff A* does
  - Limit on expansion: f_cost
  - Space complexity: linear in depth of goal node
  - Caveat: could take O(n²) time – e.g., TSP (n = 10⁶ could still be a problem)
  - Possible fix
    - Increase f cost limit by ε on each iteration
    - Approximation error bound: no worse than ε-bad (ε-admissible)
- Simplified Memory-Bounded A* – Chakrabarti, Russell (p. 102-104)
  - Idea: make space on queue as needed (compare: virtual memory)
  - Selective forgetting: drop nodes (select victims) with highest f
**Best-First Search Problems [1]: Global vs. Local Search**

- **Optimization-Based Problem Solving as Function Maximization**
  - Visualize function space
    - Criterion (z axis)
    - Solutions (x-y plane)
  - **Objective**: maximize criterion subject to
    - Solution spec
    - Degrees of freedom

- **Foothills aka Local Optima**
  - *aka* relative minima (of error), relative maxima (of criterion)
  - Qualitative description
    - All applicable operators produce suboptimal results (i.e., neighbors)
    - **However, solution is not optimal!**
  - **Discussion**: Why does this happen in optimization?

---

<table>
<thead>
<tr>
<th>d</th>
<th>IDS</th>
<th>$A'(h_1)$</th>
<th>$A'(h_2)$</th>
<th>IDS</th>
<th>$A'(h_1)$</th>
<th>$A'(h_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10</td>
<td>6</td>
<td>6</td>
<td>2.45</td>
<td>1.79</td>
<td>1.79</td>
</tr>
<tr>
<td>4</td>
<td>112</td>
<td>13</td>
<td>12</td>
<td>2.87</td>
<td>1.45</td>
<td>1.45</td>
</tr>
<tr>
<td>6</td>
<td>680</td>
<td>20</td>
<td>18</td>
<td>2.73</td>
<td>1.24</td>
<td>1.24</td>
</tr>
<tr>
<td>8</td>
<td>6384</td>
<td>39</td>
<td>25</td>
<td>2.80</td>
<td>1.33</td>
<td>1.24</td>
</tr>
<tr>
<td>10</td>
<td>47127</td>
<td>93</td>
<td>39</td>
<td>2.79</td>
<td>1.38</td>
<td>1.22</td>
</tr>
<tr>
<td>12</td>
<td>364015</td>
<td>227</td>
<td>73</td>
<td>2.78</td>
<td>1.42</td>
<td>1.24</td>
</tr>
<tr>
<td>14</td>
<td>539</td>
<td>113</td>
<td>-</td>
<td>-</td>
<td>1.44</td>
<td>1.23</td>
</tr>
<tr>
<td>16</td>
<td>1301</td>
<td>331</td>
<td>-</td>
<td>-</td>
<td>1.45</td>
<td>1.25</td>
</tr>
<tr>
<td>18</td>
<td>3056</td>
<td>363</td>
<td>-</td>
<td>-</td>
<td>1.46</td>
<td>1.26</td>
</tr>
<tr>
<td>20</td>
<td>7276</td>
<td>676</td>
<td>-</td>
<td>-</td>
<td>1.47</td>
<td>1.27</td>
</tr>
<tr>
<td>22</td>
<td>15094</td>
<td>1219</td>
<td>-</td>
<td>-</td>
<td>1.48</td>
<td>1.28</td>
</tr>
<tr>
<td>24</td>
<td>39135</td>
<td>1641</td>
<td>-</td>
<td>-</td>
<td>1.48</td>
<td>1.26</td>
</tr>
</tbody>
</table>

Figure 4.8: Comparison of the search costs and effective branching factors for the Iterative-Deepening-Search and A* algorithms with $h_1$, $h_2$. Data are averaged over 100 instances of the 8-puzzle, for various solution lengths.

Inventing admissible heuristic functions
**Best-First Search Problems [2]**

- **Lack of Gradient aka Plateaux**
  - Qualitative description
    - All neighbors indistinguishable
    - According to evaluation function $f$
  - Related problem: jump discontinuities in function space
  - Discussion: *When does this happen in heuristic problem solving?*

- **Single-Step Traps aka Ridges**
  - Qualitative description: unable to move along steepest gradient
  - Discussion: *How might this problem be overcome?*
Plan Interviews:
Week of 14 Sep 2009

- 10-15 Minute Meeting
- Discussion Topics
  - Background resources
  - Revisions needed to project plan
  - Literature review: bibliographic sources
  - Source code provided for project
  - Evaluation techniques
  - Interim goals
  - Your timeline
- Dates and Venue
  - Week of Mon 14 Sep 2009
  - Sign up for times by e-mailing CIS730TA-L@listserv.ksu.edu
- Come Prepared
  - Hard copy of plan draft
  - Screenshots or running demo for existing system you are building on
    - Installed on notebook if you have one
    - Remote desktop (RDP), VNC, or SSH otherwise
    - Link sent to CIS730TA-L@listserv.ksu.edu before interview

Project Topics Redux:
Synopsis

- Topic 1: Game-Playing Expert System
  - Angband Borg: APWborg
  - Other RPG/strategy: TIETL (http://tr.im/y7kX) / Wargus (Warcraft II clone)
  - Other games: University of Alberta GAMES (http://tr.im/y7lc)
- Topic 2: Trading Agent Competition (TAC)
  - SCM
  - Classic
- Topic 3: Data Mining – Machine Learning and Link Analysis
  - Bioinformatics: link prediction and mining, ontology development
  - Social networks: link prediction and mining
  - Other: KDDcup (http://www.sigkdd.org/kddcup)
- Topic 4: Natural Language Processing and Information Extraction
  - Machine translation
  - Named entity recognition
  - Conversational agents
- Topic 5: Computer Vision Applications
Instructions for Project Plans

- **Note:** Project Plans Are *Not* Proposals!
  - Subject to (one) revision
  - Choose one topic among three

Plan Outline: 1-2 Pages

1. Problem Statement
   - Objectives
   - Scope

2. Background
   - Related work
   - Brief survey of existing agents and approaches

3. Methodology
   - Data resources
   - Tentative list of algorithms to be implemented or adapted

4. Evaluation Methods

5. Milestones

6. References

---

Project Calendar for CIS 530 and CIS 730

- Plan Drafts – send by Fri 11 Sep 2009 (soft deadline, but by Monday)
- Plan Interviews – Mon 14 Sep 2009 – Wed 16 Sep 2009
- Revised Plans – submit by Fri 18 Sep 2009 (hard deadline)
- Interim Reports – submit by 18 Oct 2009 (hard deadline)
- Interim Interviews – around 19 Oct 2009
- Final Reports – Wed 03 Dec 2009 (hard deadline)
- Final Interviews – around Fri 05 Dec 2009
**TERMINOLOGY**

- **Properties of Search**
  - **Soundness**: returned candidate path satisfies specification
  - **Completeness**: finds path if one exists
  - **Optimality**: (usually means) achieves maximal online path cost
  - **Optimal efficiency**: (usually means) maximal offline cost

- **Heuristic Search Algorithms**
  - **Properties of heuristics**
    - Monotonicity (consistency)
    - Admissibility
  - **Properties of algorithms**
    - Admissibility (soundness)
    - Completeness
    - Optimality
    - Optimal efficiency

---

**SUMMARY POINTS**

- **Heuristic Search Algorithms**
  - **Properties of heuristics**: monotonicity, admissibility, completeness
  - **Properties of algorithms**: (soundness), completeness, optimality, optimal efficiency
  - **Iterative improvement**
    - Hill-climbing
    - Beam search
    - Simulated annealing (SA)
  - **Function maximization** formulation of search
  - **Problems**
    - Ridge
    - Foothill aka local (relative) optimum aka local minimum (of error)
    - Plateau, jump discontinuity
  - **Solutions**
    - Macro operators
    - Global optimization (genetic algorithms / SA)

- **Constraint Satisfaction Search**