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

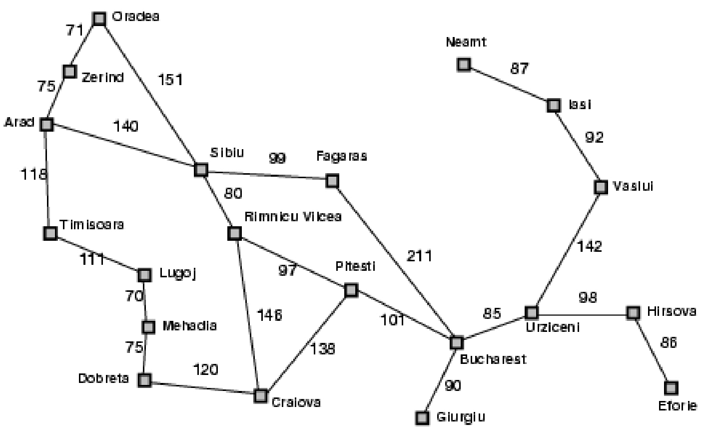
Lecture Outline

- Reading for Next Class: Section 4.3, R&N 2e
- 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

- <u>function</u> *General-Search* (*problem, strategy*) <u>returns</u> a solution <u>or</u> failure
  - Queue: represents search <u>frontier</u> (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 (<u>uniform-cost</u>, <u>bidirectional</u>), DFS (<u>depth-limited</u>, <u>ID-DFS</u>)
  - Heuristic search
    - ⇨ Based on *h* – (heuristic) function, returns estimate of min cost to goal
    - ⇨ *h* only: <u>greedy</u> (*aka* <u>myopic</u>) informed search
    - ⇨ *A/A*: f(n) = g(n) + h(n)* – <u>frontier</u> based on *estimated + accumulated* cost
- Today: More Heuristic Search Algorithms
  - A* extensions: iterative deepening (IDA*), <u>simplified memory-bounded</u> (SMA*)
  - Iterative improvement: hill-climbing, MCMC (simulated annealing)
  - Problems and solutions (macros and global optimization)

Graph Search Example: Romanian Map Revisited



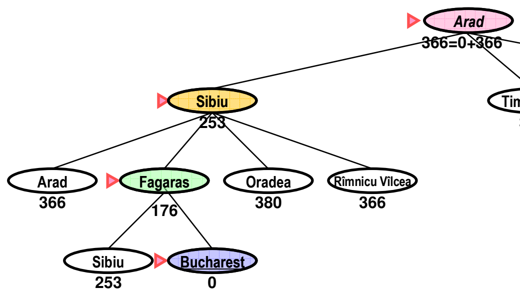
Straight–line distance to Bucharest

Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

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.

CLOSED List	OPEN List
∅ ≡ {}	*Arad*$_{366}$
Arad	Sibiu$_{253}$ Timisoara$_{32}$ Zerind$_{374}$
Arad Sibiu	Fagaras$_{176}$ T^{9}_{329} RV_{366} A_{366} Z_{374} O_{380}
Arad Sibiu Fagaras	Bucharest$_{0}$ S_{253} T_{329} RV_{366} A_{366} Z_{374} O_{380}
Arad Sibiu Fagaras Bucharest	

Path found: (*Arad* $\xrightarrow{140}$ Sibiu $\xrightarrow{99}$ Fagaras $\xrightarrow{211}$ Bucharest)$_{450}$

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?

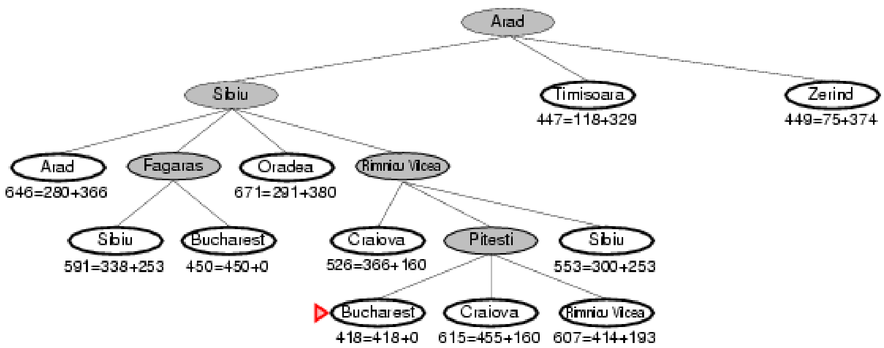
Greedy Search [4]: More Properties

- **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



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

Nodes visited (closed): {A, S, F, RV, P, B}

Path found: (*Arad* $\xrightarrow{140}$ Sibiu $\xrightarrow{80}$ Rîmnicu Vîlcea $\xrightarrow{97}$ Pitesti $\xrightarrow{99}$ Bucharest)$_{416}$

ALGORITHM A/A* [3]: PROPERTIES

- **Completeness (p. 100 R&N 2^{e})**
  - Expand lowest-cost node on fringe
  - Requires *Insert* function to insert into increasing order
- **Optimality (p. 99-101 R&N 2^{e})**
- **Optimal Efficiency (p. 97-99 R&N 2^{e})**
  - 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 2^{e})**
  - 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*: $h(m) \geq h(n) - c(n, m)$
  - Change in *h* is less than true cost
  - Intuitive idea: "No node looks artificially distant from a goal"
  - Discussion questions
    - Admissibility $\Rightarrow$ monotonicity? Monotonicity $\Rightarrow$ 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* $\forall n \,.\, h(n) \le h^*(n)$*, A* will never return a suboptimal goal node.*
  - Proof
    - Suppose A^* returns x such that $\exists\, s \,.\, g(s) < g(x)$
    - Let path from root to s be $< n_0, n_1, \ldots, n_k >$ where $n_k \equiv s$
    - Suppose A^* expands a subpath $< n_0, n_1, \ldots, n_j >$ of this path
    - Lemma: by induction on j, $s = n_k$ is expanded as well

      Base case: n_0 (root) always expanded

      Induction step: $h(n_{j+1}) \le h^*(n_{j+1})$, so $f(n_{j+1}) \le f(x)$, 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 2e)**
  - 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 2e)**
- **Iterative Deepening A* – Pearl, Korf (p. 101, R&N 2e)**
  - 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 $O(n^2)$ *time* – e.g., TSP ($n = 10^6$ 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*

	Search Cost			Effective Branching Factor		
d	IDS	$A^*(h_1)$	$A^*(h_2)$	IDS	$A^*(h_1)$	$A^*(h_2)$
2	10	6	6	2.45	1.79	1.79
4	112	13	12	2.87	1.48	1.45
6	680	20	18	2.73	1.34	1.30
8	6384	39	25	2.80	1.33	1.24
10	47127	93	39	2.79	1.38	1.22
12	3644035	227	73	2.78	1.42	1.24
14	–	539	113	–	1.44	1.23
16	–	1301	211	–	1.45	1.25
18	–	3056	363	–	1.46	1.26
20	–	7276	676	–	1.47	1.27
22	–	18094	1219	–	1.48	1.28
24	–	39135	1641	–	1.48	1.26

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 [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?***

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?***

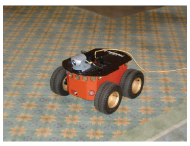
Project Topic 5 of 5: Topics in Computer Vision

Autonomous Mars Rover (Artist's Conception)

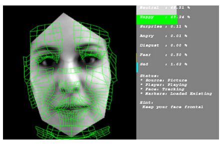
Binocular Stereo Microscopy

Edge Detection &Segmentation
Li, 2005 http://tr.im/y7d1

Image Processing



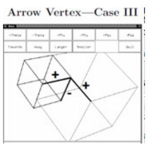
KSU Willie (Pioneer P3AT)
Gustafson et al., 2007 http://tr.im/y75U



Emotion Recognition
Gevers & Sebe, 2007 http://tr.im/y7bi

Scene Classification

Kansas State University



Waltz Line Labeling
Siskind, 2009 http://tr.im/y7ae

Hollywood Human Actions Dataset
Laptev *et al.* (2008) http://lear.inrialpes.fr

Line Labeling and Scene Understanding

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: TIELT (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**