

CSCE 580: Introduction to AI

Lecture 8: Search – Informed, Heuristics

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

12TH SEP 2024

Carolinian Creed: “I will practice personal and academic integrity.”

Credits: Copyrights of all material reused acknowledged

Organization of Lecture 8

- Introduction Segment
 - Recap of Lecture 7
 - Review of Quiz1
- Main Segment
 - Informed Search
 - Heuristics and Properties
 - Designing Heuristics
- Concluding Segment
 - Course Project Discussion
 - About Next Lecture – Lecture 8
 - Ask me anything

Introduction Section

Recap of Lecture 7

- Goal-directed problem solving agents
- How to formulate problem formulations
- Search concepts
 - Problems of controlled robot navigation, 8-tile, N-queens
- Search strategies
- Quiz 1

Review of Quiz 1

States

- Initial state
- Goal state

Actions

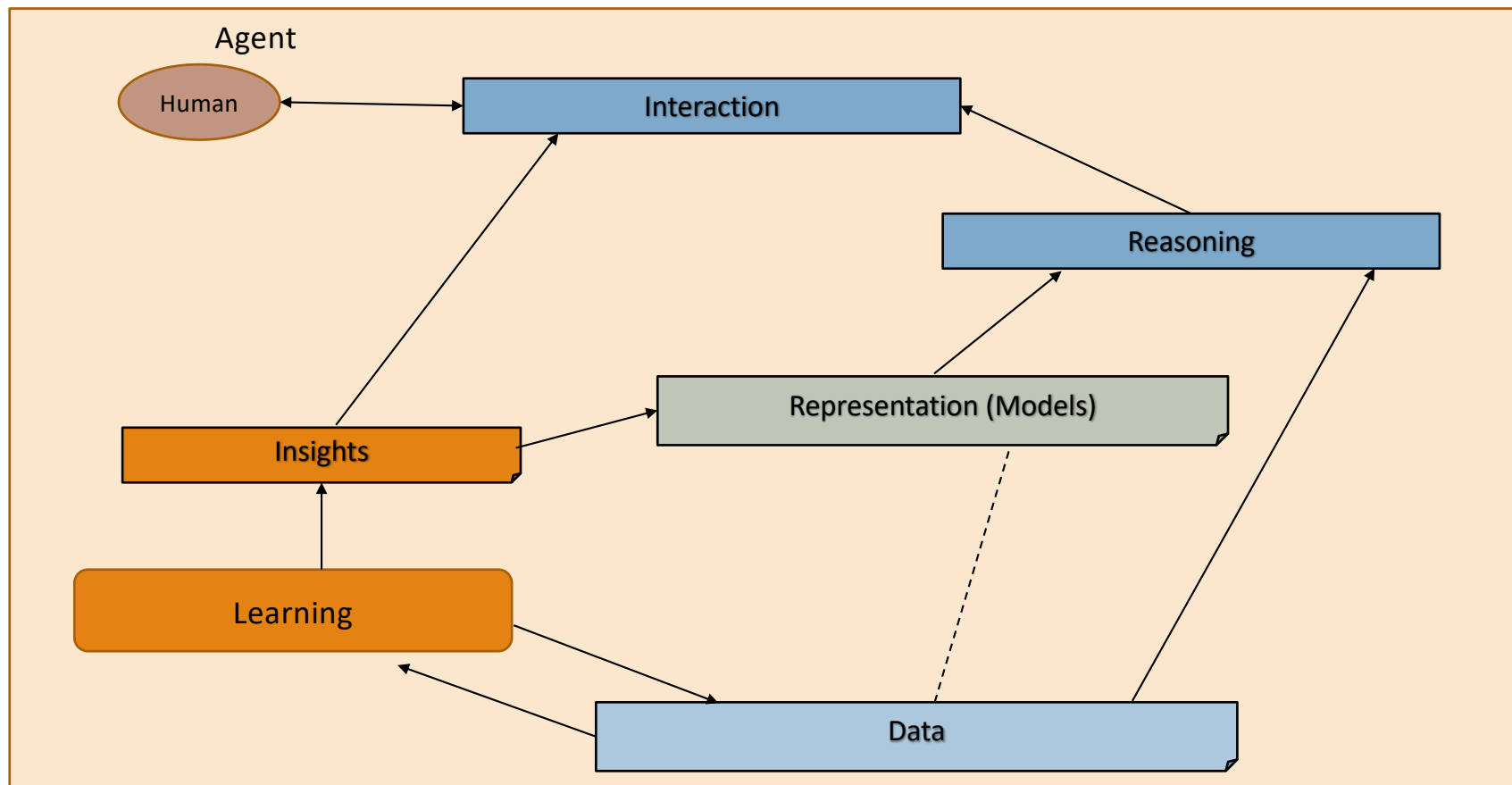
- Transition model
- Action cost

- Transition model: tells what it means to take actions -- how to move from one state to another. So, you will have to come up with a state representation, then decide on actions, and then define function(s) to state what it means to take actions.

Intelligent Agent Model



Relationship Between Main AI Topics



Where We Are in the Course

CSCE 580/ 581 – In This Course

- Week 1: Introduction, Aim: Chatbot / Intelligence Agent
- Weeks 2-3: Data: Formats, Representation and the Trust Problem
- Week 4-5: Search, Heuristics - Decision Making
- Week 6: Constraints, Optimization – Decision Making
- Week 7: Classical Machine Learning – Decision Making, Explanation
- Week 8: Machine Learning - Classification
- Week 9: Machine Learning - Classification – Trust Issues and Mitigation Methods
- Topic 10: Learning neural network, deep learning, Adversarial attacks
- Week 11: Large Language Models – Representation, Issues
- Topic 12: Markov Decision Processes, Hidden Markov models - Decision making
- Topic 13: Planning, Reinforcement Learning – Sequential decision making
- Week 14: AI for Real World: Tools, Emerging Standards and Laws; Safe AI/ Chatbots

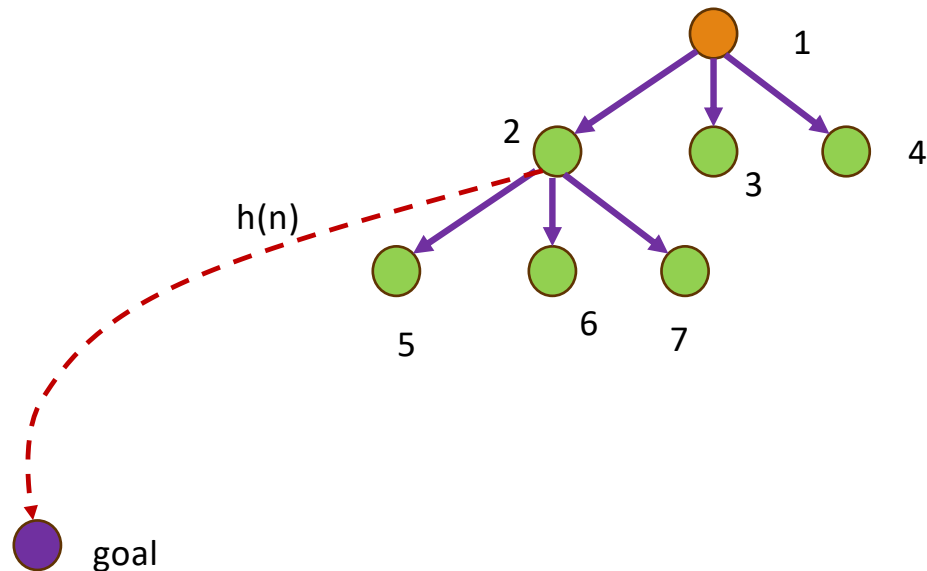
Main Section

Informed Search – Greedy best-first

Uses domain/problem specific hints to guide search

$$f(n) = h(n)$$

- f : estimated cost of best path via n to goal
- h : estimated cost to goal from n // h is also called heuristic function

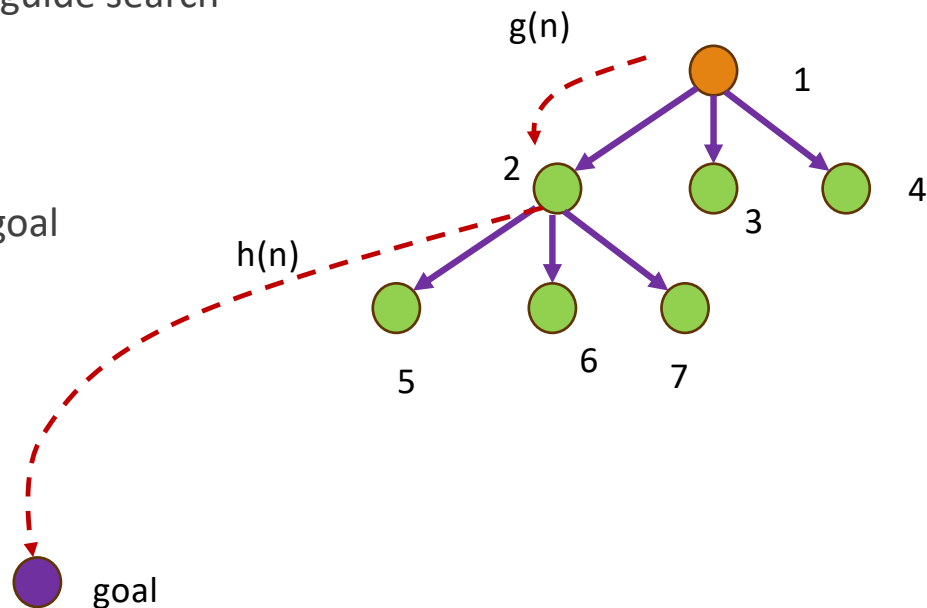


Informed Search – A* search

Uses domain/problem specific hints to guide search

$$f(n) = g(n) + h(n)$$

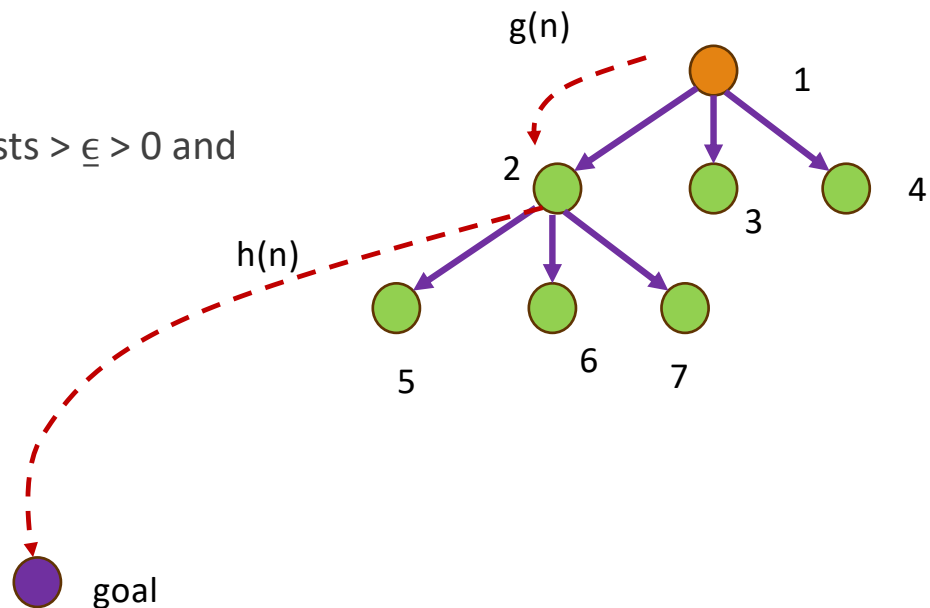
- f: estimated cost of best path via n to goal
- g: cost of best path to n
- h: estimated cost to goal from n



Properties of A* search

$$f(n) = g(n) + h(n)$$

- **A* is complete**, assuming actions costs $> \epsilon > 0$ and state space has solution or is finite
- **$h(n)$ is admissible**, i.e., never overestimates true cost to reach goal



Finding Heuristics Function

- h1: number of misplaced tiles (excluding blank)
 - $H1(\text{start}) = 8$
- h2: sum of the distance of tiles from goal (excluding blank)
 - $h2(\text{start}) = (3 + 1 + 2) + (2 + 3) + (2 + 2 + 3) = 18$
- True cost: 26

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

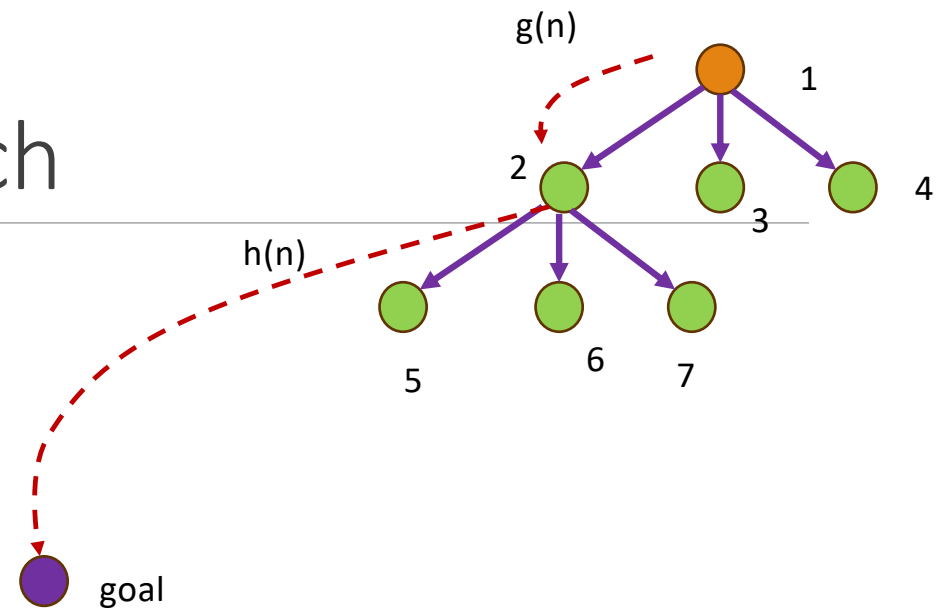
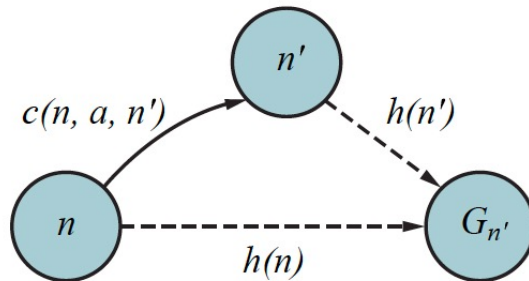
Goal State

Adapted from:
Russell & Norvig, AI: A Modern Approach

Properties of A* search

$$f(n) = g(n) + h(n)$$

- A heuristic is consistent if $h(n) \leq c(n, a, n') + h(n')$



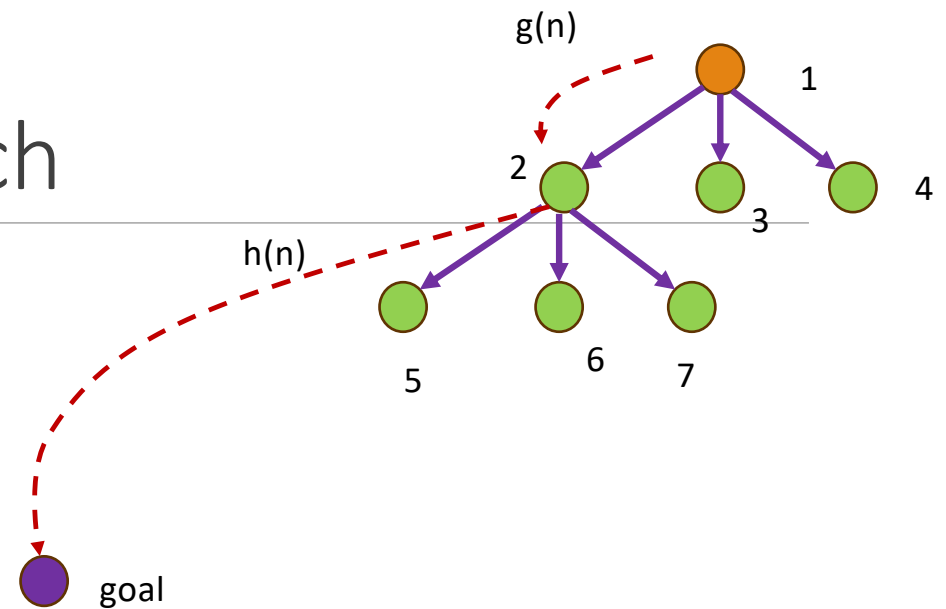
Question: with a 'random' heuristic function be consistent?

Adapted from:
Russell & Norvig, AI: A Modern Approach

Properties of A* search

$$f(n) = g(n) + h(n)$$

- A* with consistent heuristic is **optimally efficient**
- A*
 - Any algo using search path and same heuristics as A* will at least expand these nodes
 - Prunes (removes) search nodes that are not necessary for finding optimal solution

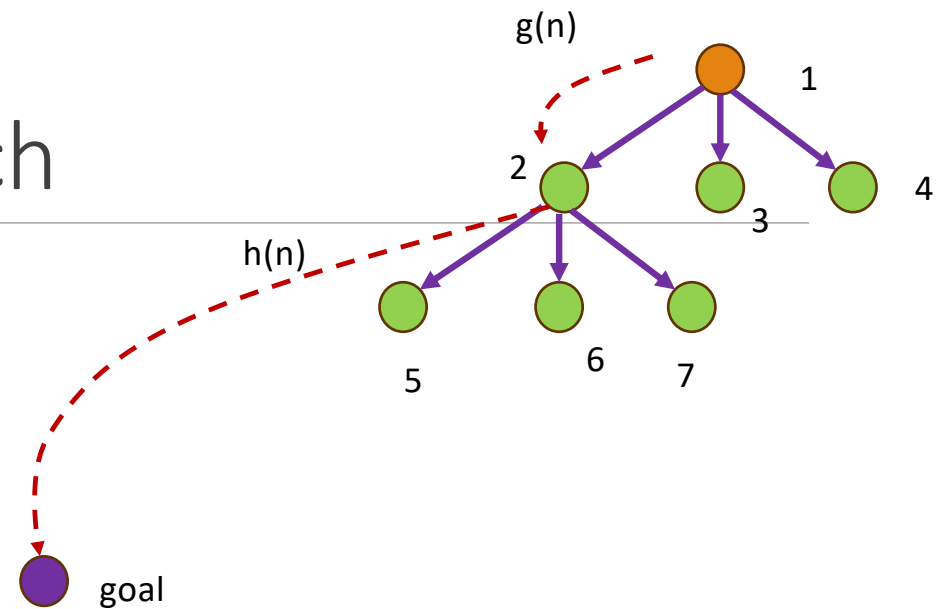


Adapted from:
Russell & Norvig, AI: A Modern Approach

Type: Satisficing Search

$$f(n) = g(n) + W * h(n)$$

- If heuristic is inadmissible, A* may find just any solution

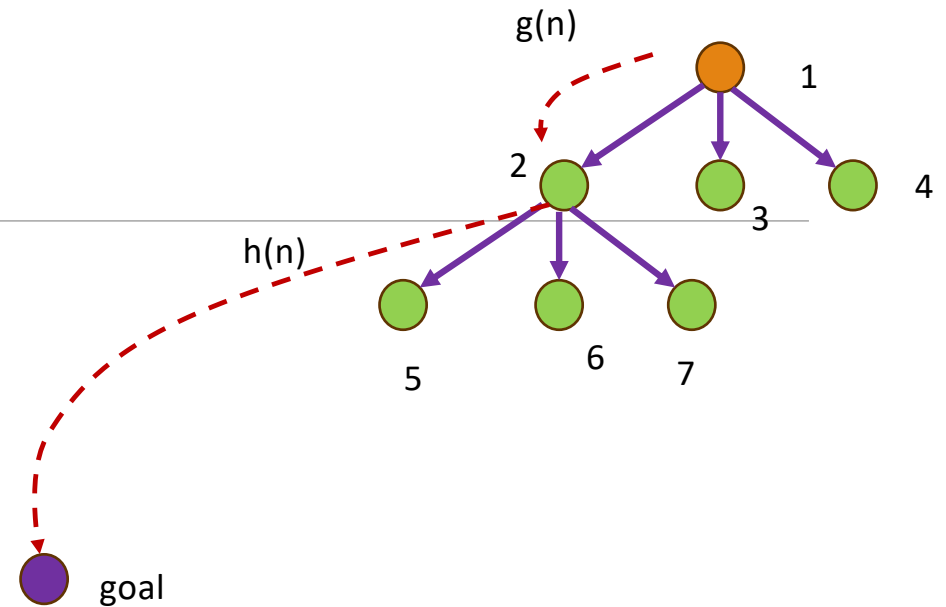


Adapted from:
Russell & Norvig, AI: A Modern Approach

Type: Beam Search

$$f(n) = g(n) + h(n)$$

- Keep only k (*a parameter*) nodes with the best f -score in frontier
- **Incomplete and sub-optimal,**
but **space efficient**

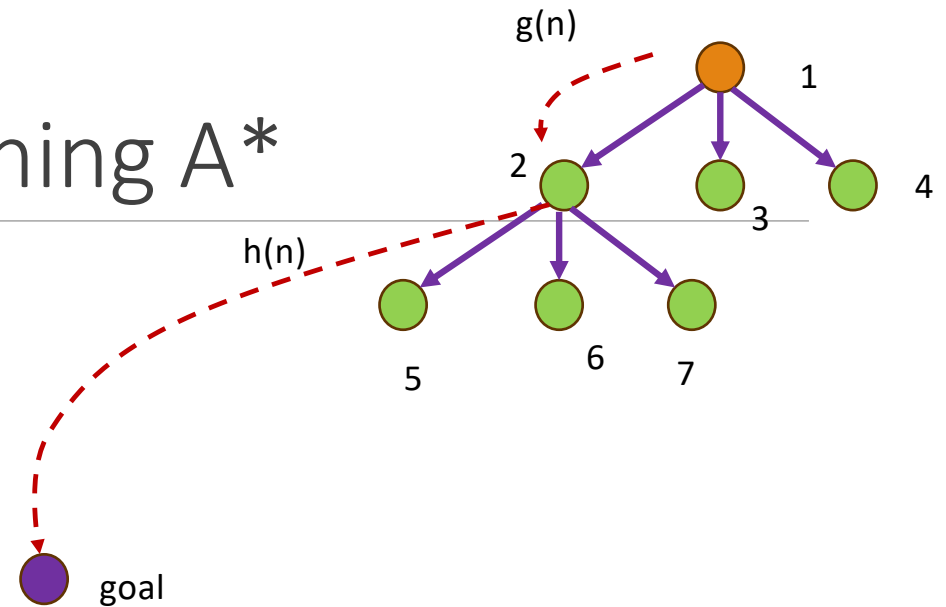


Adapted from:
Russell & Norvig, AI: A Modern Approach

Type: Iterative Deepening A*

$$f(n) = g(n) + h(n)$$

- Similar to Iterative Deepening Depth search, but for f-score. **Optimizes memory usage.**
- In each iteration, search until find a node with f-score exceeding threshold; use the node's f-score as the new threshold
- **Iterative search takes more time than plain A*. (Why?)**



Adapted from:
Russell & Norvig, AI: A Modern Approach

Informed Search Types

A* search	$f(n) = g(n) + h(n)$	(W = 1)
Uniform-cost search	$f(n) = g(n)$	(W = 0)
Greedy best-first search	$f(n) = h(n)$	(W = 1)
Weighted A* search	$f(n) = g(n) + W * h(n)$	(1 < W < infinite)

Notes:

Uniform-cost => uninformed

Weighted A* => satisficing

Impact of Heuristics Function

d	Search Cost (nodes generated)			Effective Branching Factor		
	BFS	$A^*(h_1)$	$A^*(h_2)$	BFS	$A^*(h_1)$	$A^*(h_2)$
6	128	24	19	2.01	1.42	1.34
8	368	48	31	1.91	1.40	1.30
10	1033	116	48	1.85	1.43	1.27
12	2672	279	84	1.80	1.45	1.28
14	6783	678	174	1.77	1.47	1.31
16	17270	1683	364	1.74	1.48	1.32
18	41558	4102	751	1.72	1.49	1.34
20	91493	9905	1318	1.69	1.50	1.34
22	175921	22955	2548	1.66	1.50	1.34
24	290082	53039	5733	1.62	1.50	1.36
26	395355	110372	10080	1.58	1.50	1.35
28	463234	202565	22055	1.53	1.49	1.36

Reduces effective branching factor!

Figure 3.26 Comparison of the search costs and effective branching factors for 8-puzzle problems using breadth-first search, A^* with h_1 (misplaced tiles), and A^* with h_2 (Manhattan distance). Data are averaged over 100 puzzles for each solution length d from 6 to 28.

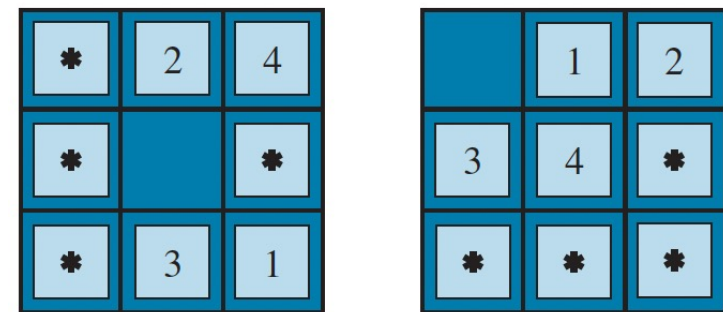
Adapted from:
Russell & Norvig, AI: A Modern Approach

Choosing From a Choice of (Admissible) Heuristics

- Choose dominating heuristics
 - For n , $h_2(n) \geq h_1(n)$
- If not dominating, choose maximum
 - $h(n) = \max \{h_1(n), h_2(n), \dots, h_k(n)\}$

Creating Heuristics Automatically

- From relaxed problems
 - Formulate a relaxed problem
 - Solve relaxed problem
 - Use solution length as heuristics for original problem
(Relaxed problem heuristics)
- From sub-problems
 - Formulate a sub-problem
 - Solve relaxed sub-problem
 - Store solution of sub-problem
 - Compute admissible heuristic h_{DB} for each node by looking up sub-problem and its solution cost
(Pattern databases)
- Learn heuristics
 - From data: past solutions, relaxed problems, ...
 - **Predict heuristic value**



Start State

Goal State

Adapted from:
Russell & Norvig, AI: A Modern Approach

Coding Example

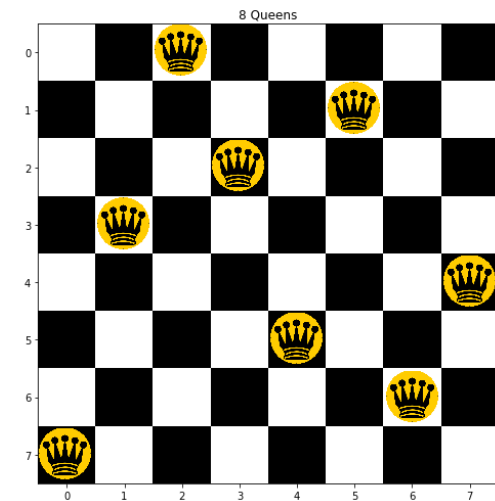
- 8-Puzzle – code notebook
 - <https://github.com/biplav-s/course-ai-tai-f23/blob/main/sample-code/Class6-To-Class10-search.md>
- From AIMA book
 - <https://github.com/aimacode/aima-python/blob/master/search.ipynb>
 - See 8-tile example

Discussion: Relaxed Problems

- For N-Queens
- Pancake problem

- Many more

<https://www.movingai.com/SAS/index.html>



Adapted from:
Russell & Norvig, AI: A Modern Approach

Discussion: Rubic's Cube

- Search and deep-learning
 - Demo video: Solving with search and distance-based heuristics
<https://youtu.be/YQZ2sj-x5js>
 - Live demo: Solving with A*search and deep learning-based heuristics (DeepCube-A)
<https://deepcube.igb.uci.edu/>

Uninformed Search Strategies

Search strategies use only the information available in the problem definition. They do not use a measure of distance to goal (uninformed).

- Breadth-first search
- Uniform-cost search
- Depth-first search
- Depth-limited search
- Iterative deepening search
- Bidirectional search

Consideration: type of queue used for the **fringe of the search tree**
(collection of tree nodes that have been generated but not yet expanded)

Adapted from:

1. Russell & Norvig, AI: A Modern Approach
2. Bart Selman's CS 4700 Course

Informed Search – A* search

- Best-first
- A*
- Weighted A*
- Beam search [Incomplete]
- Iterative-deepening A* [Incomplete]

A* search	$f(n) = g(n) + h(n)$	(W = 1)
Uniform-cost search	$f(n) = g(n)$	(W = 0)
Greedy best-first search	$f(n) = h(n)$	(W = 1)
Weighted A* search	$f(n) = g(n) + W * h(n)$	(1 < W < infinite)

Lecture 8: Summary

- We talked about
 - Informed Search
 - Heuristics and Properties
 - Designing Heuristics

Concluding Section

Course Project

Discussion: Projects

- New: two projects
 - Project 1: model assignment
 - Project 2: single problem/ llm based solving / fine-tuning/ presenting result

Project Discussion

1. Go to Google spreadsheet against your name
2. Enter model assignment name and link from (<http://modelai.gettysburg.edu/>)

1. Create a private Github repository called “CSCE58x-Fall2024-<studentname>-Repo”. Share with Instructor (biplav-s) and TA (vishalpallagani)
2. Create Google folder called “CSCE58x-Fall2024-<studentname>-SharedInfo”. Share with Instructor (prof.biplav@gmail.com) and TA (vishal.pallagani@gmail.com)
3. Create a Google doc in your Google repo called “Project Plan” and have the following by next class (Sep 5, 2024)

Timeline

1. Title:
2. Key idea: (2-3 lines)
3. Data need:
4. Methods:
5. Evaluation:
6. Milestones
 1. // Create your own
7. Oct 3, 2024

Reference: Project 1 Rubric (30% of Course)

Assume total for Project-1 as 100

- **Project results** – 60%
 - Working system ? – 30%
 - Evaluation with results superior to baseline? – 20%
 - Went through project tasks completely ? – 10%
- **Project efforts** – 40%
 - Project report – 20%
 - Project presentation (updates, final) – 20%
- **Bonus**
 - Challenge level of problem – 10%
 - Instructor discretion – 10%
- **Penalty**
 - Lack of timeliness as per your milestones policy (right) - up to 30%

Milestones and Penalties

- Project plan due by Sep 5, 2024 [-10%]
- Project deliverables due by Oct 3, 2024 [-10%]
- Project presentation on Oct 8, 2024 [-10%]

About Next Lecture – Lecture 9

Lecture 9: Local Search

- Searching in large spaces
 - Hill climbing
 - Simulated Annealing
 - Genetic programming
- Class 10: Adversarial games and search