SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Heuristics

PROF LIM KWAN HUI

50.021 Artificial Intelligence

The following notes are compiled from various sources such as textbooks, lecture materials, Web resources and are shared for academic purposes only, intended for use by students registered for a specific course. In the interest of brevity, every source is not cited. The compiler of these notes gratefully acknowledges all such sources.

Outline & Objectives

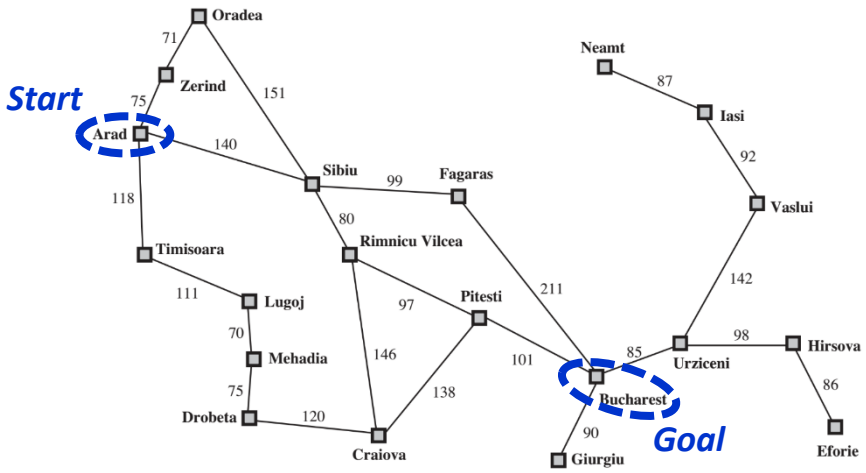
- Understand the role that heuristic plays in informed search
- Understand the important properties of heuristics
  - Admissible, Consistent, Dominance
- Able to design heuristics that are appropriate for specific problem instances, based on these properties
  - Relax problem
  - Sub-problem

Recap: UCS vs Greedy Best-First Search

- What are the issues with UCS, in contrast to ?
  - Complete and optimal but may “waste” search in the wrong direction
- What are the issues with greedy best-first search?
  - Search generally in the “right” direction but not complete or optimal
- Is there a way to combine the two and address each’s shortcomings?
  - UCS: $f(n) = g(n)$
  - Greedy: $f(n) = h(n)$
  - Combined: $f(n) = g(n) + h(n)$ → ???

Recap: A* Search

- General idea: Expand the node *n* that has *incurred the least cost* and is *nearest to the goal state*
- Implementation: Using a *priority* queue ordered by *eval. func. f(n)*
  - Evaluation function *f(n) = g(n) + h(n)*
  - Path cost g(n) = total path cost from start node to node *n*
  - Heuristic h(n) = estimated distance from node *n* to goal state



Straight Line Distances to Bucharest

Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

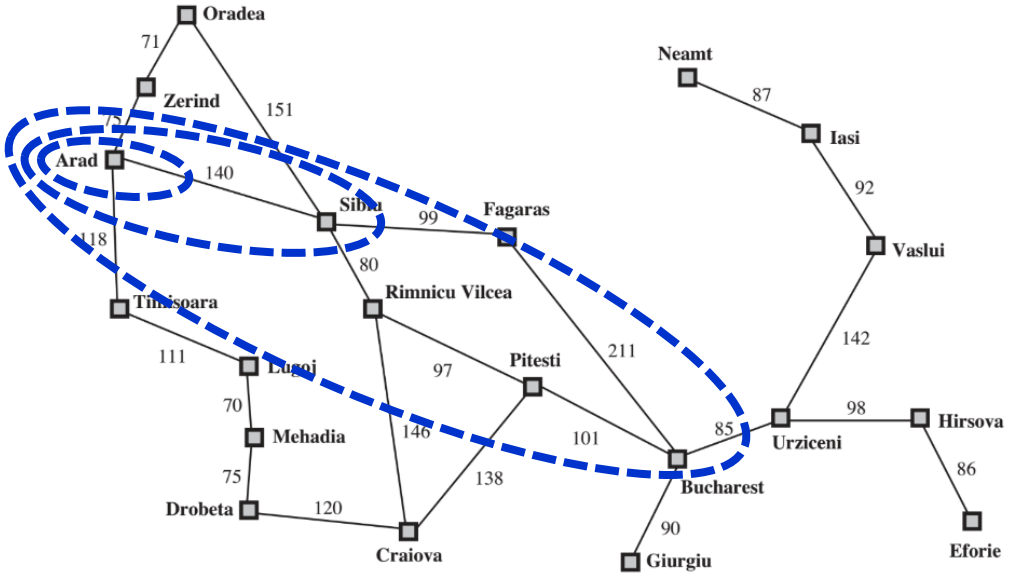
Recap: Properties of A* Search

- Completeness: Yes
- Optimality: Yes

  (If heuristics are admissible/consistent, more on this later)
- Time complexity: Same as UCS
- Space complexity: Same as UCS

Optimality of A* (intuitive)

- Lemma: A* search expands nodes in order of increasing *f value*
- Gradually adds "*f-contours*" of nodes
- Contour *i* has all nodes with $f=f_i$, where $f_i < f_{i+1}$



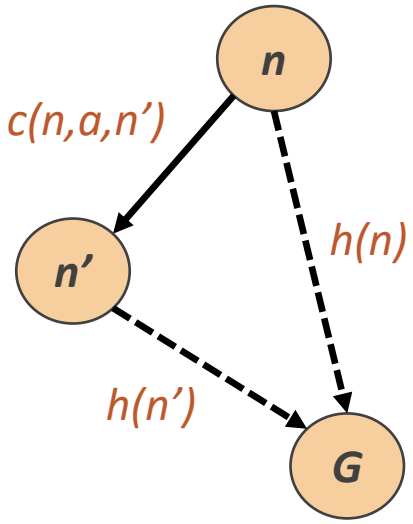
Property: Admissible Heuristic

- A heuristic *h(n)* is *admissible* if $h(n) \leq h^*(n)$
  - *h(n)* = estimated distance from node *n* to goal state
  - $h^*(n)$ = true cost from node *n* to goal state

- Example: Romania holiday
  - *h(n)* = straight line distance from current city to destination city
  - $h^*(n)$ = actual road distance from current city to destination city
  - In this case, the straight line distance can never be greater than the actual road distance

Property: Consistent Heuristic

- A heuristic *h(n)* is *consistent* if *h(n) ≤ c(n,a,n') + h(n')*
  - *h(n)* = estimated distance from node *n* to goal state *G*
  - *h(n')* = estimated distance from node *n'* to goal state *G*
  - *c(n,a,n')* = cost of getting from node *n* to *n'*



Property: Dominance

- A heuristic $h_2(n)$ *dominates* $h_1(n)$ if $h_2(n) \geq h_1(n)$, for all n
  - Provided that both heuristics are admissible

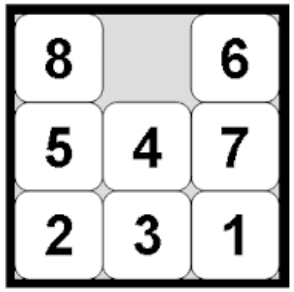
- A more dominant heuristic will be better for search
  - Potentially explore less branches

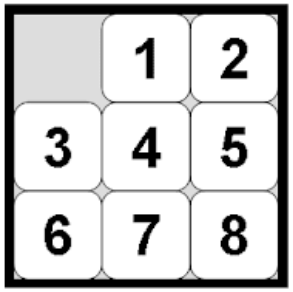
Property: Dominance

- A heuristic $h_2(n)$ *dominates* $h_1(n)$ if $h_2(n) \geq h_1(n)$
  - Provided that both heuristics are admissible
- A more dominant heuristic will be better for search
  - Potentially explore less branches
- How do we design heuristics?

Recap: 8-puzzle

Initial State



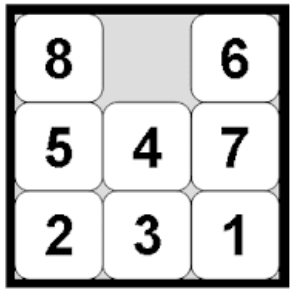


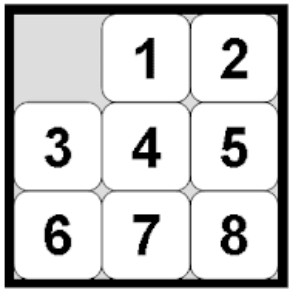
Goal State

- **State space**: Number tiles in each cell position
- **Initial state**: [8,-,6,5,4,7,2,3,1]
- **Actions**: Move tile *{Left, Right, Up, Down}*
- **Transition Model**: Update tiles in current and target cell positions
- **Path cost**: Number of moves
- **Goal test**: Compare to positions in goal state

Exercise: Designing Heuristics

Initial State



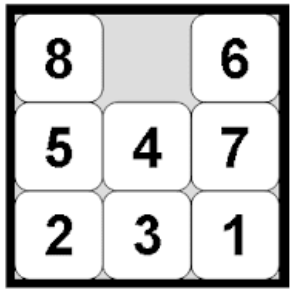


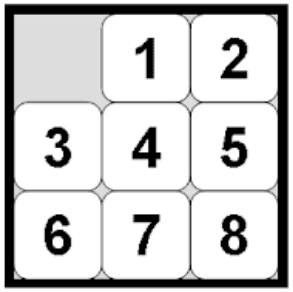
Goal State

- Relaxed problems
  - Admissible heuristics can be derived from the *exact solution cost* of a *relaxed version* of the problem
  - E.g., Planning Romania holiday via roads. Relaxed: assuming direct paths
- How about heuristics for the 8-puzzle?

Exercise: Designing Heuristics

Initial State





Goal State

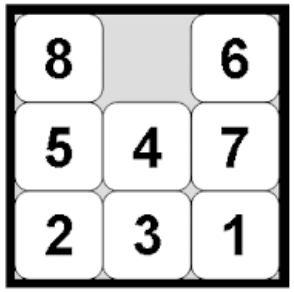
- Relaxed problem
  - Tile can be directly moved to any square

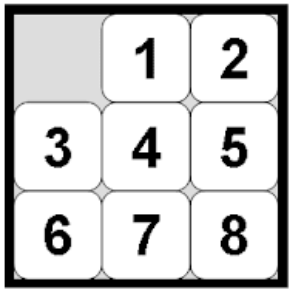
- $h_1(n)$ = number of misplaced tiles

  = ?

Exercise: Designing Heuristics

Initial State





Goal State

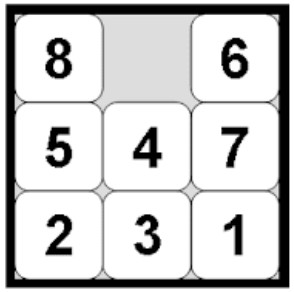
- Relaxed problem
  - Tile can be directly moved to any square

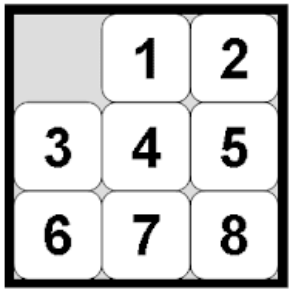
- $h_1(n)$ = number of misplaced tiles

  = 7

Exercise: Designing Heuristics

Initial State





Goal State

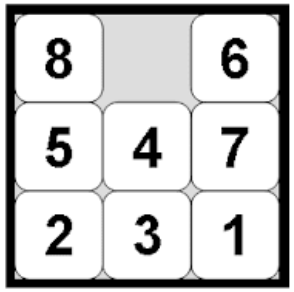
- Relaxed problem
  - Tile can be moved to any adjacent square

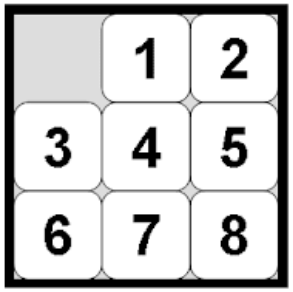
- $h_2(n)$ = total Manhattan distance

  = ?

Exercise: Designing Heuristics

Initial State





Goal State

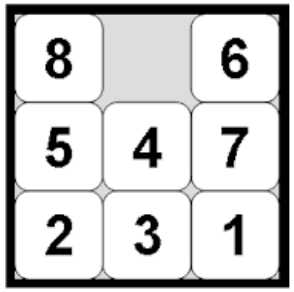
- Relaxed problem
  - Tile can be moved to any adjacent square

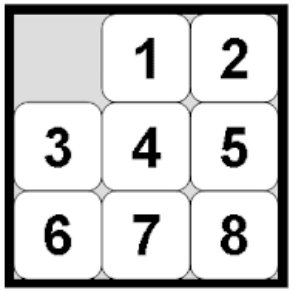
- $h_2(n)$ = total Manhattan distance

  = 3+4+2+0+2+4+2+4 = 21

Exercise: Designing Heuristics

Initial State



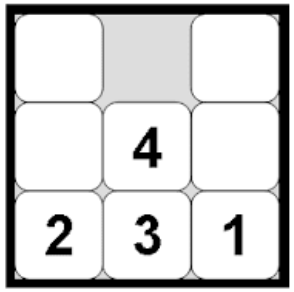


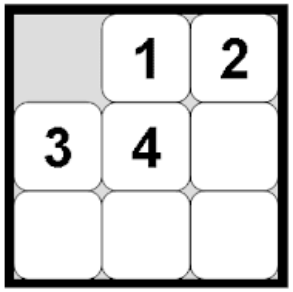
Goal State

- Sub-problems
  - Admissible heuristics can be derived from the *solution cost* of a *sub-problem* of the problem

Exercise: Designing Heuristics

Initial State



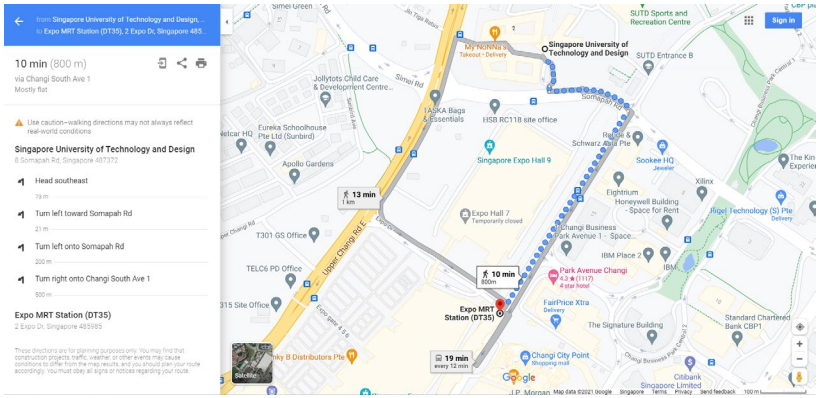


Goal State

- Sub-problems
  - Admissible heuristics can be derived from the *solution cost* of a *sub-problem* of the problem
  - E.g., Instead of solving for all 8 tiles, solve for a sub-problem with 4 tiles

A* Search Applications

- Path finding problems
- Video games
- Resource planning problems
- Robot motion planning



Summary & Objectives

- Understand the role that heuristic plays in informed search
- Understand the important properties of heuristics
  - Admissible, Consistent, Dominance
- Able to design heuristics that are appropriate for specific problem instances, based on these properties
  - Relax problem
  - Sub-problem