CS3243 : Introduction to Artificial Intelligence

Tutorial 3

NUS School of Computing

July 7, 2022

Admin

- Consultations
- Extra material for practice

Review

- Heuristics, more importantly on how do we come up with them
- Dominance and how it helps
- Relaxed problems, the recipe for obtaining heuristics

Review

Admissibility	h is admissible iff $\forall n : h(n) \leq h*(n)$ An admissible heuristic will never overestimate the cost to reach the goal
Consistency	h is consistent, iff for every node n and every successor n' of n generated by action a $h(n) \leq c(n, a, n') + h(n')$
Dominance	$h_1(n)$ dominates $h_2(n)$ iff $\forall n : h_1(n) \geq h_2(n)$

Review (8-puzzle)

- Original Problem : A tile can move from square X to square Y if X is adjacent to Y and Y is blank

Review (8-puzzle)

- Original Problem : A tile can move from square X to square Y if X is adjacent to Y and Y is blank
- Relaxed Version 1 : A tile can move from square X to square Y ~~if X is adjacent to Y and Y is blank~~

Review (8-puzzle)

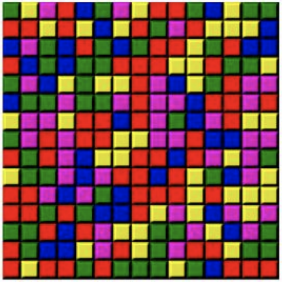
- Original Problem : A tile can move from square X to square Y if X is adjacent to Y and Y is blank
- Relaxed Version 1 : A tile can move from square X to square Y ~~if X is adjacent to Y and Y is blank~~
- Relaxed Version 2 : A tile can move from square X to square Y if X is adjacent to Y ~~and Y is blank~~

Tutorial Question 1

- Design Admissible Heuristics for SameGame Puzzle
- Given information :
  - Initial state : Grid with $n \times m$ tiles with c different colours
  - Neighbors : Directly adjacent tiles (Internal tiles : 4 neighbors, Edge tiles : 3 neighbors, Corner tiles : 2 neighbors)
  - Group : Set of ≥ 2 neighboring tiles of same color
  - Singleton : Tile not belonging to any group
  - Action/Move : Deleting a group (not singleton). Thereafter, vertical gravity and column shifting (right-to-left) applies (in transition model)
  - Goal state : Empty grid (assume solvable)
  - Transition cost : 1 (or ∞ if no groups exist)

Tutorial Question 1

- An example of the initial state :



- Top-left corner is $(1,1)$, $1 \leq ROW \leq n$, and $1 \leq COL \leq m$

Tutorial Question 1

- Idea - Admissible : $\forall n,\ h(n) \leq h^*(n)$
- We have two approaches

Tutorial Question 1

- Idea - Admissible : $\forall n,\ h(n) \leq h^*(n)$
- We have two approaches
- Approach 1 : By reasoning/inference (ie What information could I use at each state to get a estimate of the number of moves needed to reach the goal? Important : Must be an underestimate!)

Tutorial Question 1

- Idea - Admissible : $\forall n,\ h(n) \leq h^*(n)$
- We have two approaches
- Approach 1 : By reasoning/inference (ie What information could I use at each state to get a estimate of the number of moves needed to reach the goal? Important : Must be an underestimate!)
- Approach 2 (better approach) : Relaxation of the game rules (ie. I now need less moves than what I originally need to reach the goal because the game has become "easier" $=$ admissible. Important : We should be able to easily obtain the number of moves needed with the relaxed rules)

Tutorial Question 1

- Proof of admissibility : $h(s) =$ Number of colours remaining.
- Each group contains exactly 1 colour
- Each remaining color has 1 or more groups
- Each move may reduce groups by more than 1 (example when groups of same color combine), but will never reduce remaining groups of same color below 1 (unless that color is removed by the move)
- Hence, each legal move will leave the number of colors remaining unchanged or reduce it by at most 1 if the color is removed by the move, ie. $h(s) \leq h^*(s)$

Tutorial Question 1

- Proof of admissibility : $h(s) =$ Number of colours remaining.
- Each group contains exactly 1 colour
- Each remaining color has 1 or more groups
- Each move may reduce groups by more than 1 (example when groups of same color combine), but will never reduce remaining groups of same color below 1 (unless that color is removed by the move)
- Hence, each legal move will leave the number of colors remaining unchanged or reduce it by at most 1 if the color is removed by the move, ie. $h(s) \leq h^*(s)$
- Note : You may also prove admissibility by showing that heuristic was derived from a relaxed version of the game (Please refer to the forum post on LumiNUS for details)

Tutorial Question 3

- Given information :
  - State : Position of Pac-Man in the grid at any point in time and the positions of the remaining (i.e., uneaten) pellets.
  - Initial state: Grid entirely filled with pellets except at Pac-Man's starting position.
  - Action: Move up/down/left/right.
  - Transition model: Updating the position of Pac-Man given current grid and action, and eating (ie. removing) the pellet at this new position if a pellet exists. Returns updated grid.
  - Goal state: No (uneaten) pellets left in the grid.
  - Transition/Step cost: 1 for each action taken.

Tutorial Question 3

- Let's say admissible heuristics : h_1, h_3, h_4
- On to comparing the dominance of each on another

Tutorial Question 3

- h_1 *vs* h_3
- What do you think?

Tutorial Question 3

- h_1 *vs* h_3
- What do you think?
- Let's look at the counter-examples

Tutorial Question 3

- h_1 *vs* h_4
- What do you think?

Tutorial Question 3

- h_1 *vs* h_4
- What do you think?
- Let's look at the previous counter-examples

Tutorial Question 3

- h_3 *vs* h_4
- What do you think?

Tutorial Question 3

- h_3 *vs* h_4
- What do you think?
- Average $\leq$ Maximum, and Euclidean distance $\leq$ Manhattan distance; hence Average Euclidean distance (h_4) $\leq$ Maximum Manhattan distance (h_3) for all nodes ie h_3 dominates h_4

Tutorial Question 4

- Traveling Salesman Problem (TSP)
- Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once, and returns to the origin city?

Tutorial Question 4

- Traveling Salesman Problem (TSP)
- Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once, and returns to the origin city?
- Now, this is a hard problem. We haven't been able to find a polynomial time algorithm for it. But, we have heuristics to the rescue.

Tutorial Question 4(a)

- How can MST heuristic be derived from a relaxed version of TSP?

Tutorial Question 4(a)

- How can MST heuristic be derived from a relaxed version of TSP?
- The TSP problem is to find a minimal length path through the cities, which forms a closed loop.
- MST is a relaxed version of the TSP problem because it only seeks for a minimal length graph that need not be a closed loop – it can be any fully-connected graph.

Tutorial Question 4(b)

- Is the MST heuristic admissible?

Tutorial Question 4(b)

- Is the MST heuristic admissible?
- Yes, because it is always shorter than or equal to the length of a closed loop through the cities.

Tutorial Question 4(c)

- Hill Climbing Approach to solve TSP?

Tutorial Question 4(c)

- Hill Climbing Approach to solve TSP?
- Algorithm:
  - Choose a random path that connects all the cities
  - Along that path, pick two cities at random
  - Partition the whole path across these two cities, which leaves us with 3 paths
  - Try all 6 possible permutations of arranging the partitions to obtain the least cost loop
  - Upon getting the best cost, repeat the steps to further improve
  - Halt when done with a certain k number of iterations

Thank you!

If you have any questions, please don't hesitate. Feel free to ask!
We are here to learn together!