CS 188
Fall 2021

Search and CSPs Solutions

Q1. Power Pellets

Consider a Pacman game where Pacman can eat 3 types of pellets:

- Normal pellets (n-pellets), which are worth one point.
- Decaying pellets (d-pellets), which are worth $max(0, 5 - t)$ points, where t is time.
- Growing pellets (g-pellets), which are worth t points, where t is time.

The location and type of each pellet is fixed. The pellet's point value stops changing once eaten. For example, if Pacman eats one g-pellet at $t = 1$ and one d-pellet at $t = 2$, Pacman will have won $1 + 3 = 4$ points.

Pacman needs to find a path to win at least 10 points but he wants to minimize distance travelled. The cost between states is equal to distance travelled.

(a) Which of the following must be including for a minimum, sufficient state space?

[✓] Pacman's location
☐ Location and type of each pellet
☐ How far Pacman has travelled
[✓] Current time
☐ How many pellets Pacman has eaten and the point value of each eaten pellet
[✓] Total points Pacman has won
[✓] Which pellets Pacman has eaten

A state space should include which pellets are left on the board, the current value of pellets, Pacman's location, and the total points collected so far. With this in mind:
(1) The starting location and type of each pellet are not included in the state space as this is something that does not change during the search. This is analogous to how the walls of a Pacman board are not included in the state space.
(2) How far Pacman has travelled does not need to be explicitly tracked by the state, since this will be reflected in the cost of a path.
(3) Pacman does need the current time to determine the value of pellets on the board.
(4) The number of pellets Pacman has eaten is extraneous.
(5) Pacman must track the total number of points won for the goal test.
(6) Pacman must know which pellets remain on the board, which is the complement of the pellets he has eaten.

(b) Which of the following are admissible heuristics? Let x be the number of points won so far.

[✓] Distance to closest pellet, except if in the goal state, in which case the heuristic value is 0.
☐ Distance needed to win $10 - x$ points, determining the value of all pellets as if they were n-pellets.
Distance needed to win $10 - x$ points, determining the value of all pellets as if they were g-pellets (i.e. all pellet values will be t.)
☐ Distance needed to win $10 - x$ points, determining the value of all pellets as if they were d-pellets (i.e. all pellet values will be $max(0, 5 - t)$.
☐ Distance needed to win $10 - x$ points assuming all pellets maintain current point value (g-pellets stop

increasing in value and d-pellets stop decreasing in value)

☐ None of the above

(1) Admissible; to get 10 points Pacman will always have to travel at least as far as the distance to the closest pellet, so this will always be an underestimate.
(2) Not admissible; if all the pellets are actually g-pellets, assuming they are n-pellets will lead to Pacman collecting more pellets in more locations, and thus travel further.
(3) Not admissible; if pellets are n-pellets or g-pellets, Pacman would have an overestimate.
(4) Not admissible; if pellets are g-pellets, then using the current pellet value might lead Pacman to collect more locations, and thus travel further than necesarry.

(c) Instead of finding a path which minimizes distance, Pacman would like to find a path which minimizes the following:

$$C_{new} = a * t + b * d$$

where t is the amount of time elapsed, d is the distance travelled, and a and b are non-negative constants such that $a + b = 1$. Pacman knows an admissible heuristic when he is trying to minimize time (i.e. when $a = 1, b = 0$), h_t, and when he is trying to minimize distance, h_d (i.e. when $a = 0, b = 1$).
Which of the following heuristics is guaranteed to be admissible when minimizing C_{new}?

☐ $mean(h_t, h_d)$ [✓ $min(h_t, h_d)$] ☐ $max(h_t, h_d)$ [✓ $a * h_t + b * h_d$]
☐ None of the above

For this question, think about the inequality $C_{new} = a * t + b * d \geq a * h_t + b * h_d$. We can guarantee a heuristic h_{new} is admissible if $h_{new} \leq a * h_t + b * h_d$
(1) If $a = b$, $0.5 * h_t + 0.5 * h_d$ is not guaranteed to be less than $a * h_t + b * h_d$, so this will not be admissible.
(2) $min(h_t, h_d) = a * min(h_t, h_d) + b * min(h_t, h_d) \leq a * h_t + b * h_d$
(3) $max(h_t, h_d)$ will be greater than $a * h_t + b * h_d$ unless $h_t = h_d$, wo this will not be admissible.
(4) Admissible.

Q2. Rubik's Search

A Rubik's cube has about 4.3×10^{19} possible configurations, but any configuration can be solved in 20 moves or less. We pose the problem of solving a Rubik's cube as a search problem, where the states are the possible configurations, and there is an edge between two states if we can get from one state to another in a single move. Thus, we have 4.3×10^{19} states. Each edge has cost 1. Since we can make 27 moves from each state, the branching factor is 27. Since any configuration can be solved in 20 moves or less, we have $h^*(n) \leq 20$.

For each of the following searches, estimate the approximate number of states expanded. Mark the option that is closest to the number of states expanded by the search. Assume that the shortest solution for our start state takes exactly 20 moves. Note that 27^{20} is much larger than 4.3×10^{19}.

(a) DFS Tree Search

(i) Best Case: [✓ 20] ○ 4.3×10^{19} ○ 27^{20} ○ ∞ (never finishes)

(ii) Worst Case: ○ 20 ○ 4.3×10^{19} ○ 27^{20} [✓ ∞ (never finishes)]

(b) DFS graph search

(i) Best Case: [✓ 20] ○ 4.3×10^{19} ○ 27^{20} ○ ∞ (never finishes)

(ii) Worst Case: ○ 20 [✓ 4.3×10^{19}] ○ 27^{20} ○ ∞ (never finishes)

(c) BFS tree search

(i) Best Case: ○ 20 ○ 4.3×10^{19} [✓ 27^{20}] ○ ∞ (never finishes)

(ii) Worst Case: ○ 20 ○ 4.3×10^{19} [✓ 27^{20}] ○ ∞ (never finishes)

(d) BFS graph search

(i) Best Case: ○ 20 [✓ 4.3×10^{19}] ○ 27^{20} ○ ∞ (never finishes)

(ii) Worst Case: ○ 20 [✓ 4.3×10^{19}] ○ 27^{20} ○ ∞ (never finishes)

(e) A* tree search with a perfect heuristic, h*(n), Best Case

[✓ 20] ○ 4.3×10^{19} ○ 27^{20} ○ ∞ (never finishes)

(f) A* tree search with a bad heuristic, h(n) = 20 - h*(n), Worst Case

○ 20 ○ 4.3×10^{19} [✓ 27^{20}] ○ ∞ (never finishes)

(g) A* graph search with a perfect heuristic, h*(n), Best Case

[✓ 20] ○ 4.3×10^{19} ○ 27^{20} ○ ∞ (never finishes)

(h) A* graph search with a bad heuristic, h(n) = 20 - h*(n), Worst Case

○ 20 [✓ 4.3×10^{19}] ○ 27^{20} ○ ∞ (never finishes)

Q3. CSPs: Potluck Pandemonium

The potluck is coming up and the staff haven't figured out what to bring yet! They've pooled their resources and determined that they can bring some subset of the following items.

1. Pho
2. Apricots
3. Frozen Yogurt
4. Fried Rice
5. Apple Pie
6. Animal Crackers

There are five people on the course staff: Taylor, Jonathan, Faraz, Brian, and Alvin. Each of them will only bring one item to the potluck.

i. If (F)araz brings the same item as someone else, it cannot be (B)rian.

ii. (A)lvin has pho-phobia so he won't bring Pho, but he'll be okay if someone else brings it.

iii. (B)rian is no longer allowed near a stove, so he can only bring items 2, 3, or 6.

iv. (F)araz literally can't even; he won't bring items 2, 4, or 6.

v. (J)onathan was busy, so he didn't see the last third of the list. Therefore, he will only bring item 1, 2, 3, or 4.

vi. (T)aylor will only bring an item that is before an item that (J)onathan brings.

vii. (T)aylor is allergic to animal crackers, so he won't bring item 6. (If someone else brings it, he'll just stay away from that table.)

viii. (F)araz and (J)onathan will only bring items that have the same first letter (e.g. Frozen Yogurt and Fried Rice).

ix. (B)rian will only bring an item that is after an item that (A)lvin brings on the list.

x. (J)onathan and (T)aylor want to be unique; they won't bring the same item as anyone else.

(a) Which of the listed constraints are unary constraints?

☐ i [✓ ii] [✓ iii] [✓ iv] [✓ v]

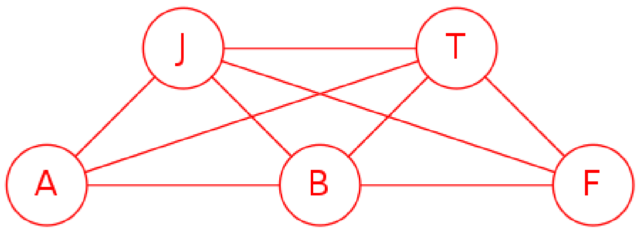
☐ vi [✓ vii] ☐ viii ☐ ix ☐ x

(b) Rewrite implicit constraint viii. as an explicit constraint.

(F, J) $\in$ { (3, 4), (4, 3),
(2, 5), (5, 2),
(2, 6), (6, 2),
(5, 6), (6, 5),
(1, 1), (2, 2), (3, 3),
(4, 4), (5, 5), (6, 6)
}

(c) How many edges are there in the constraint graph for this CSP?

There are 9 edges in this constraint graph.



(d) The table below shows the variable domains after all unary constraints have been enforced.

A		2	3	4	5	6
B		2	3			6
F	1		3		5	
J	1	2	3	4		
T	1	2	3	4	5	

Following the MRV heuristic, which variable should we assign first? Break all ties alphabetically.

○ A [✓ B] ○ F ○ J ○ T

(e) To decouple this from the previous question, assume that we choose to assign (F)araz first. In this question, we will choose which value to assign to using the Least Constraining Value method.

To determine the number of remaining values, enforce arc consistency to prune the domains. Then, count the total number of possible assignments (**not** the total number of remaining values). It may help you to enforce arc consistency twice, once before assigning values to (F)araz, and then again after assigning a value.

(i) Assigning F = 1 results in 0 possible assignments.

Assigning F = 1 leaves no possible values in J's domain (due to constraint viii).

(ii) Assigning F = 3 results in 5 possible assignments.

Assigning F = 3 leaves J's domain as $\{4\}$. Enforcing arc consistency gives A = $\{2, 3, 5\}$, B = $\{6\}$, and T = $\{1, 2\}$. Therefore, the 5 possible assignments are (A, B, F, J, T) = (2, 6, 3, 4, 1), (3, 6, 3, 4, 1), (5, 6, 3, 4, 1), (3, 6, 3, 4, 2), (5, 6, 3, 4, 2).

(iii) Assigning F = 5 results in 3 possible assignments.

Assigning F = 5 leaves J's domain as $\{2\}$. Enforcing arc consistency gives A = $\{3, 4, 5\}$, B = $\{6\}$, and T = $\{1\}$. Therefore, the 3 possible assignments are (A, B, F, J, T) = (3, 6, 5, 2, 1), (4, 6, 5, 2, 1), (5, 6, 5, 2, 1).

(iv) Using the LCV method, which value should we assign to F? If there is a tie, choose the lower number.

○ 1 ○ 2 [✓ 3] ○ 4 ○ 5 ○ 6