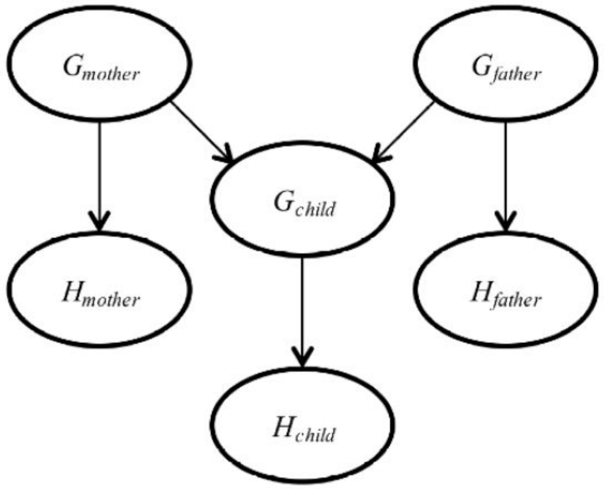
Homework Exercises

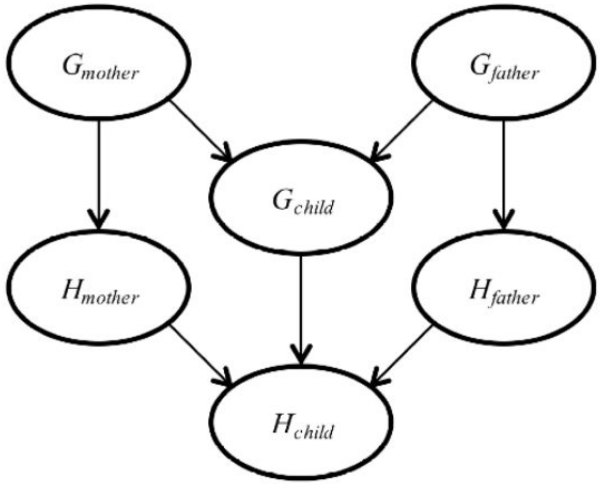
Question 6: Bayes' Nets Independence

0.0/6.0 points (graded)

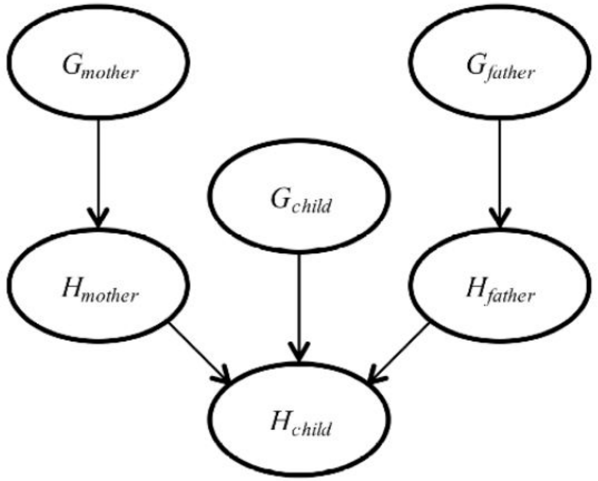
Let H_x be a random variable denoting the handedness of an individual x, with possible values l or r. A common hypothesis is that left- or right-handedness is inherited by a simple mechanism; that is, perhaps there is a gene G_x, also with values l or r, and perhaps actual handedness turns out mostly the same (with some probability s) as the gene an individual possesses. Furthermore, perhaps the gene itself is equally likely to be inherited from either of an individual's parents, with a small nonzero probability m of a random mutation flipping the handedness.



(a)



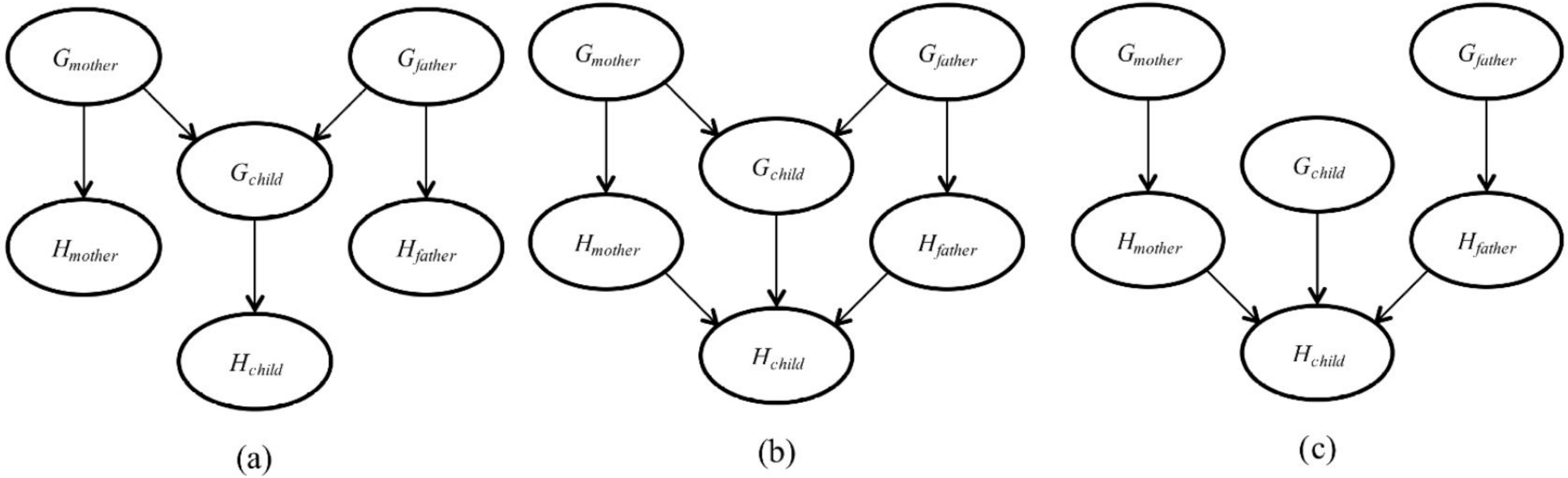
(b)

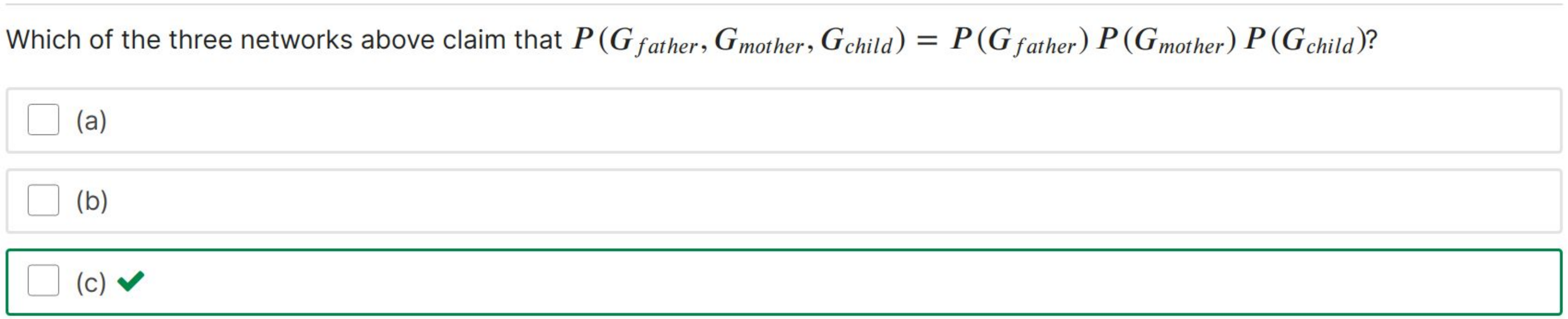


(c)

Which of the three networks above claim that $P(G_{father}, G_{mother}, G_{child}) = P(G_{father})\,P(G_{mother})\,P(G_{child})$?

- ☐ (a)
- ☐ (b)
- ☐ (c)





Which of the three networks above claim that $P(G_{father}, G_{mother}, G_{child}) = P(G_{father})\,P(G_{mother})\,P(G_{child})$?

☐ (a)

☐ (b)

☐ (c) ✓

This question asks which of the three BNs enforce the independence of G_father, G_mother, and G_child. Only (c) enforces this because of the “common effect” (aka V-shape) triplet.

Which of the three networks make independence claims that are consistent with the hypothesis about the inheritance of handedness?

- ☐ (a)
- ☐ (b)
- ☐ (c)

$$\begin{tikzcd}
G_{mother} \arrow[dd] \arrow[dr] & & G_{father} \arrow[dl] \arrow[dd] \\
& G_{child} \arrow[dd] & \\
H_{mother} & & H_{father} \\
& H_{child} &
\end{tikzcd}$$

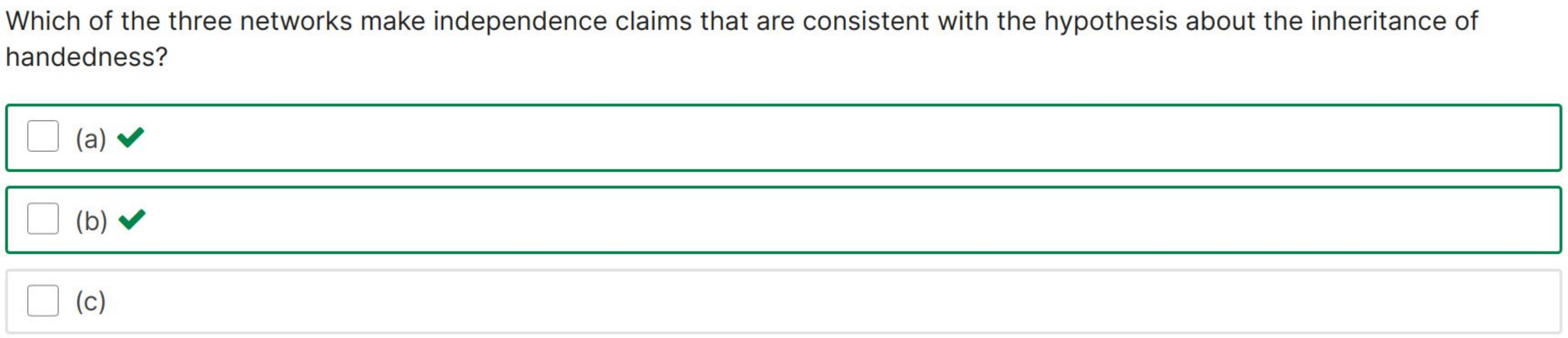
(a)

$$\begin{tikzcd}
G_{mother} \arrow[dd] \arrow[dr] & & G_{father} \arrow[dl] \arrow[dd] \\
& G_{child} \arrow[dd] & \\
H_{mother} \arrow[dr] & & H_{father} \arrow[dl] \\
& H_{child} &
\end{tikzcd}$$

(b)

$$\begin{tikzcd}
G_{mother} \arrow[dd] & & G_{father} \arrow[dd] \\
& G_{child} \arrow[dd] & \\
H_{mother} \arrow[dr] & & H_{father} \arrow[dl] \\
& H_{child} &
\end{tikzcd}$$

(c)



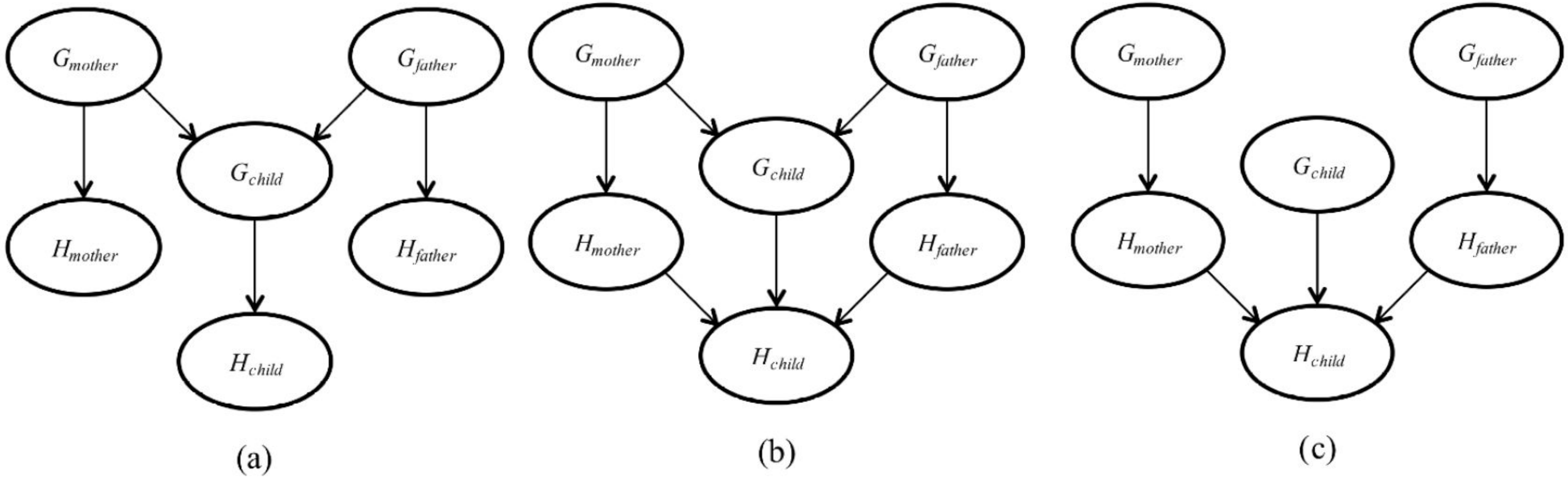
Which of the three networks make independence claims that are consistent with the hypothesis about the inheritance of handedness?

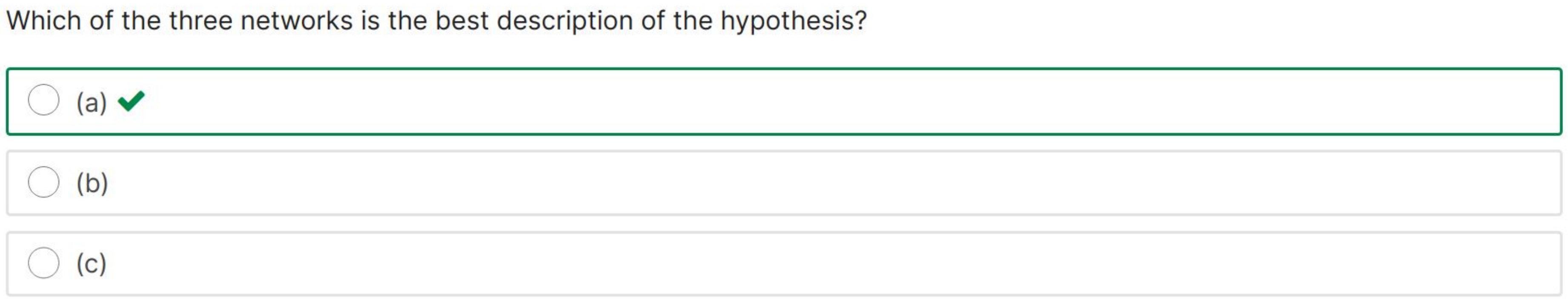
- ☐ (a) ✓
- ☐ (b) ✓
- ☐ (c)

The hypothesis is basically “gene G affects handness H” and “gene G inherits from genes of parents”. Hence, a BN that is consistent with this hypothesis must NOT enforce independence b/t a person’s gene and its handness and b/t a person’s gene and his/her parents’ genes.
Neither (a) nor (b) enforce these independences, hence the answer.

Which of the three networks is the best description of the hypothesis?

- ◯ (a)
- ◯ (b)
- ◯ (c)





Which of the three networks is the best description of the hypothesis?

- ○ (a) ✓
- ○ (b)
- ○ (c)

The hypothesis states “perhaps actual handedness turns out mostly the same (...) as the gene an individual possesses”, which hypothesizes that gene is the only influence of a person’s handness.

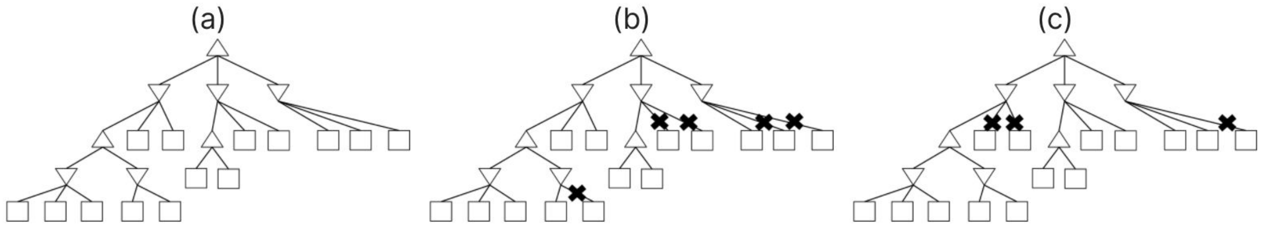
Though both (a) and (b) are consistent with the hypothesis, (b) connects handness of parents to handness of child, which suggests there is an potential influence b/t parents’ handness and child’s handedness. This makes (b) in a sense more “redundant” and hence (a) is a more accurate, more succinct description.

HW 3 (Games): Q6

Question 6: Possible Pruning

0.0/5.0 points (graded)

Assume we run α - β pruning, expanding successors from left to right, on a game with tree as shown in Figure (a) below.



☐ There exists an assignment of utilities to the terminal nodes such that no pruning will be achieved (shown in Figure (a)).

☐ There exists an assignment of utilities to the terminal nodes such that the pruning shown in Figure (b) will be achieved.

☐ There exists an assignment of utilities to the terminal nodes such that the pruning shown in Figure (c) will be achieved.

☐ None of the above.

a) One such assignment:
3,3 | 5,5 | 6,6,6
4,4
1,1,1 | 2,2

b) One such assignment:
2,2 | x,x | 0,x,x
0,0
1,1,1 | 0,x

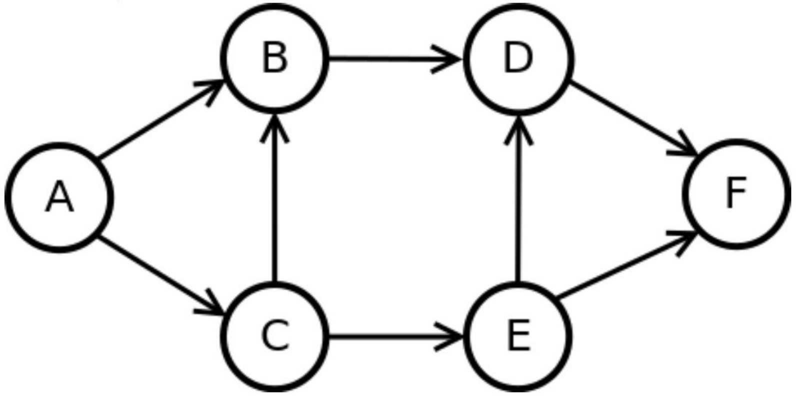
c)
The left-most child of the root would have $\alpha = -\infty$, so its direct children can never be pruned because they will always be greater than $-\infty$. Intuitively the root hasn't seen anything else yet, so any value returned by the minimizer might end up being taken by its parent.

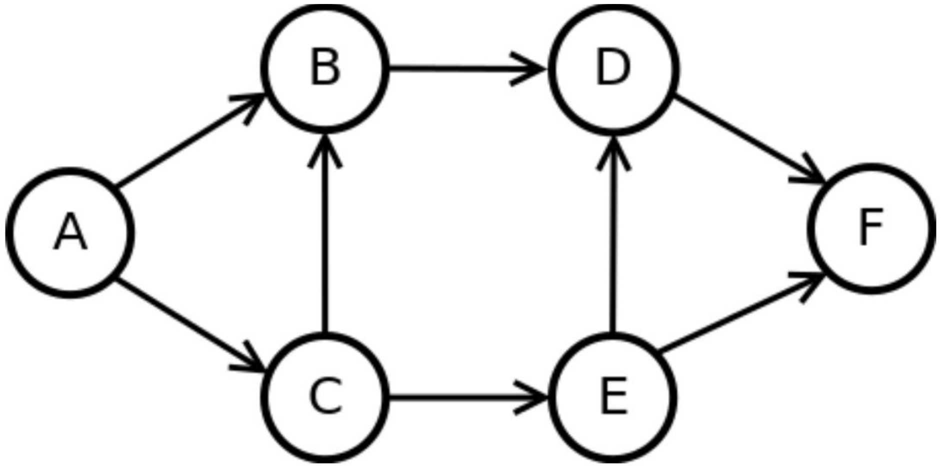
HW 4 (MDPs): Q5

Question 5.1: Value Iteration Convergence

0.0/4.0 points (graded)

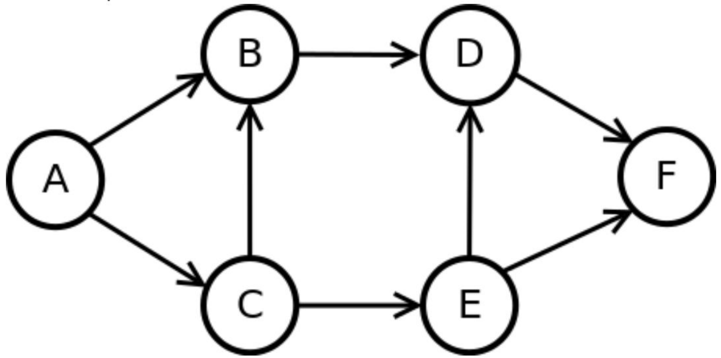
This is a randomized question. The variables that change with each reset of the question are colored **blue**.
We will consider a simple MDP that has six states, A, B, C, D, E, and F. Each state has a single action, go. An arrow from a state x to a state y indicates that it is possible to transition from state x to next state y when go is taken. If there are multiple arrows leaving a state x, transitioning to each of the next states is equally likely. The state F has no outgoing arrows: once you arrive in F, you stay in F for all future times. The reward is one for all transitions, with one exception: staying in F gets a reward of zero. Assume a discount factor = 0.5. We assume that we initialize the value of each state to 0. (Note: you should not need to explicitly run value iteration to solve this problem.)





Part 1: After how many iterations of value iteration will the value for state C have become exactly equal to the true optimum? (Enter inf if the values will never become equal to the true optimal but only converge to the true optimal.)

Part 2: How many iterations of value iteration will it take for the values of all states to converge to the true optimal values? (Enter inf if the values will never become equal to the true optimal but only converge to the true optimal.)



Part 1: After how many iterations of value iteration will the value for state C have become exactly equal to the true optimum? (Enter inf if the values will never become equal to the true optimal but only converge to the true optimal.)

Answer: 3

Part 2: How many iterations of value iteration will it take for the values of all states to converge to the true optimal values? (Enter inf if the values will never become equal to the true optimal but only converge to the true optimal.)

Answer: 4

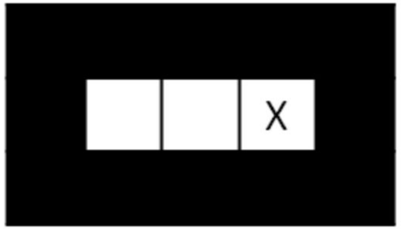
Because there are no moves from state F, we have the optimal value of F upon initializing. Since all the rewards are earned from transitions, finding the optimal value of a state amounts to finding the longest path from that state to F. For example, state D, whose longest path to F is only length 1, will find its optimal value after only one iteration.
$V^*(D) = V_1(D) = R(D, go, F) + \gamma V^*(F) = 1$
Similarly, the state A will find its optimal value after four iterations, because it will find out about its length 4 path to F after four iterations. Because A's length 4 path is the longest of the graph, it will take four iterations for all states to converge to their optimal values.

HW1 (Search): Q9

Q9: Hive Minds: Lost at Night

0.0/9.0 points (graded)

It is night and you control a single insect. You know the maze, but you do not know what square the insect will start in. You must pose a search problem whose solution is an all-purpose sequence of actions such that, after executing those actions, the insect is guaranteed to be on the exit square, regardless of initial position. The insect executes the actions mindlessly and does not know whether its moves succeed: if it uses an action which would move it in a blocked direction, it will stay where it is. For example, in the maze below, moving right twice guarantees that the insect will be at the exit regardless of its starting position.



Which of the following state representations could be used to solve this problem?

- ○ A tuple (x, y) representing the position of the insect.
- ○ A tuple (x, y) representing the position of the insect, plus a list of all squares visited by the insect.
- ○ An integer t representing how many time steps have passed, plus an integer b representing how many times the insect's motion has been blocked by a wall.
- ○ A list of boolean variables, one for each position in the maze, indicating whether the insect could be in that position.
- ○ A list of all positions the insect has been in so far.

Which of the following state representations could be used to solve this problem?

- ◯ A tuple (x, y) representing the position of the insect.
- ◯ A tuple (x, y) representing the position of the insect, plus a list of all squares visited by the insect.
- ◯ An integer t representing how many time steps have passed, plus an integer b representing how many times the insect's motion has been blocked by a wall.
- ◯ A list of boolean variables, one for each position in the maze, indicating whether the insect could be in that position. ✓
- ◯ A list of all positions the insect has been in so far.

Goal test: $\forall\,(x, y)$, if $(x, y) = Goal : bool\,[x]\,[y] = True$; else $bool\,[x]\,[y] = False$
Successor: Actions available to us are {NORTH, SOUTH, EAST, WEST}. The action WEST, for example, will move us from a state $bool$ to a new state $bool_{next}$ such that $\forall\,(x, y)$, if $bool\,[x+1]\,[y] = True$ or $(x-1, y)$ is a wall $: bool_{next}\,[x]\,[y] = True$; else $bool\,[x]\,[y] = False$.

What is the size of the state space?

- ○ MN
- ○ MNT
- ○ 2^{MN}
- ○ $(MN)^T$
- ○ e^{2MN}
- ○ The state space is infinite.

What is the size of the state space?

- ○ MN
- ○ MNT
- ○ 2^{MN} ✓
- ○ $(MN)^T$
- ○ e^{2MN}
- ○ The state space is infinite.

There are MN total booleans, and 2 values for each boolean, so there are 2^{MN} total possible states.

Which of the following are admissible heuristics?

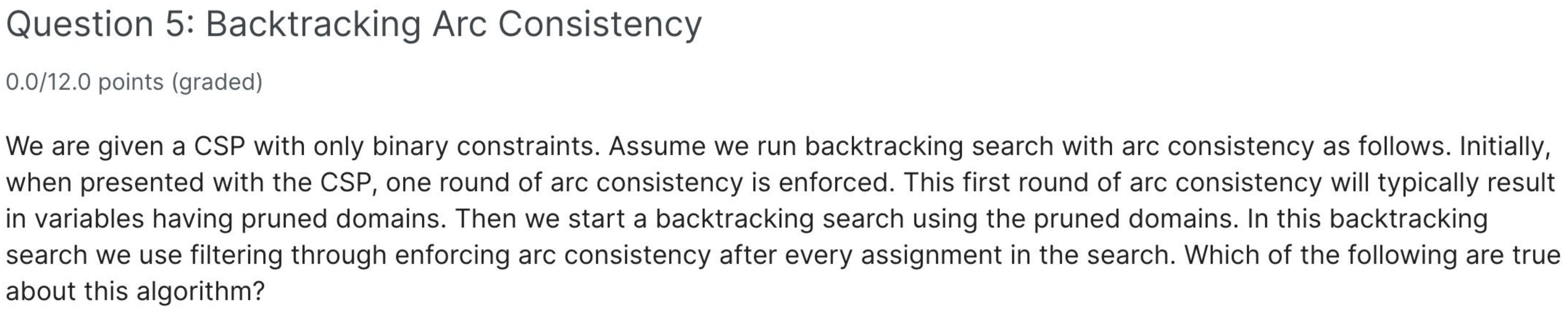
- ☐ Total number of possible locations the insect might be in.
- ☐ The maximum of Manhattan distances to the exit square from each possible location the insect could be in.
- ☐ The minimum of Manhattan distances to the exit square from each possible location the insect could be in.

Which of the following are admissible heuristics?

- ☐ Total number of possible locations the insect might be in.
- ☐ The maximum of Manhattan distances to the exit square from each possible location the insect could be in. ✓
- ☐ The minimum of Manhattan distances to the exit square from each possible location the insect could be in. ✓

Option 1: Consider the maze above. From the beginning state, we are 2 moves away from the exit square. However, the heuristic would suggest 3, which is not admissable.
Option 2: Our all-purpose solution must work for the possible insect location that is furthest from the exit square, so the solution must be cost at least as much as the cost from that location, so this heuristic is admissable.
Option 3: Admissable, for the same reason as above.

HW2 (CSPs): Q5



Question 5: Backtracking Arc Consistency

0.0/12.0 points (graded)

We are given a CSP with only binary constraints. Assume we run backtracking search with arc consistency as follows. Initially, when presented with the CSP, one round of arc consistency is enforced. This first round of arc consistency will typically result in variables having pruned domains. Then we start a backtracking search using the pruned domains. In this backtracking search we use filtering through enforcing arc consistency after every assignment in the search. Which of the following are true about this algorithm?

Part 1

Which of the following are true about this algorithm?

- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables being empty, this means the CSP has no solution.
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domain of one of the not yet assigned variables being empty, this means the CSP has no solution.
- ☐ None of the above.

Part 1

Which of the following are true about this algorithm?

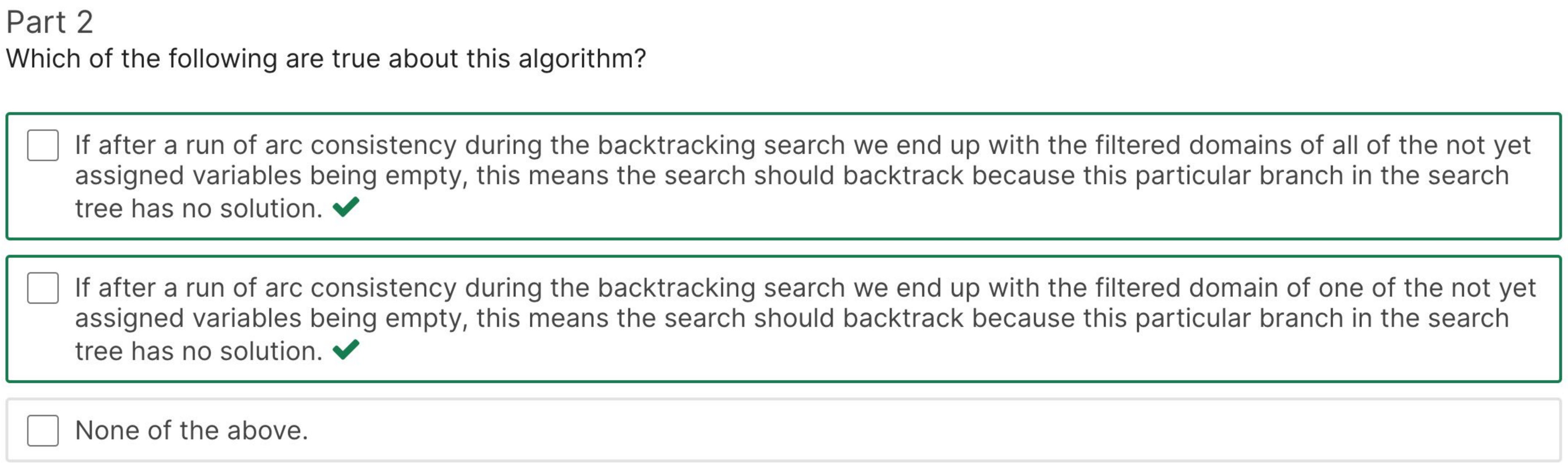
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables being empty, this means the CSP has no solution.
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domain of one of the not yet assigned variables being empty, this means the CSP has no solution.
- ☐ None of the above. ✓

Part 1:
Both of these cases mean that along the current branch of the search tree there is no solution for the csp, and that backtracking is required. In the special case where no variables have been assigned, then both of these statements are true.

Part 2

Which of the following are true about this algorithm?

- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables being empty, this means the search should backtrack because this particular branch in the search tree has no solution.
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domain of one of the not yet assigned variables being empty, this means the search should backtrack because this particular branch in the search tree has no solution.
- ☐ None of the above.



Part 2

Which of the following are true about this algorithm?

- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables being empty, this means the search should backtrack because this particular branch in the search tree has no solution. ✓
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domain of one of the not yet assigned variables being empty, this means the search should backtrack because this particular branch in the search tree has no solution. ✓
- ☐ None of the above.

Part 2:

If any domain is empty, that means that no value in that domain can satisfy all of the constraints with the currently assigned variables. This means that we have to go back and change at least one of those assignments. In the special case where no variables have been assigned, this means that the CSP has no solution.

Part 3

Which of the following are true about this algorithm?

- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables each having exactly one value left, this means we have found a solution.
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables each having more than one value left, this means we have found a whole space of solutions and we can just pick any combination of values still left in the domains and that will be a solution.
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables each having more than one value left, this means we can’t know yet whether there is a solution somewhere further down this branch of the tree, and search has to continue down this branch to determine this.

Part 3

Which of the following are true about this algorithm?

- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables each having exactly one value left, this means we have found a solution. ✓
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables each having more than one value left, this means we have found a whole space of solutions and we can just pick any combination of values still left in the domains and that will be a solution.
- ☐ If after a run of arc consistency during the backtracking search we end up with the filtered domains of all of the not yet assigned variables each having more than one value left, this means we can't know yet whether there is a solution somewhere further down this branch of the tree, and search has to continue down this branch to determine this. ✓

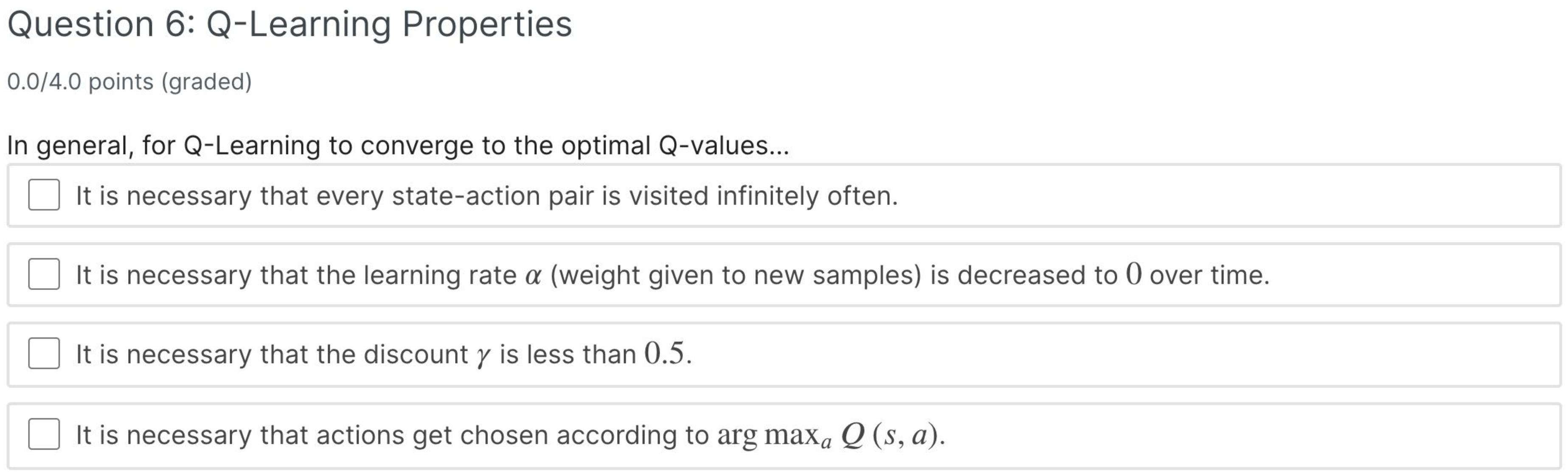
Part 3:

Option 1: If there is exactly one value left in every domain, that means that those values satisfy all of the binary constraints of the problem. Explicitly, for every variable, the assignment of the single remaining value has at least one possible value for every other variable, and one of those values is the one that remains in the domain.

Option 2: This case means that we have to continue searching. For example, consider the CSP used in earlier solutions with three variables, A, B, and C, each of which has two values in its domain {1,2}, with the only constraint being that no two variables can have the same value. Arc consistency does not filter anything out, but we don't know yet whether or not there is a solution.

Option 3: See the explanation for option 2.

HW 5 (RL): Q6



Question 6: Q-Learning Properties

0.0/4.0 points (graded)

In general, for Q-Learning to converge to the optimal Q-values...

- ☐ It is necessary that every state-action pair is visited infinitely often.
- ☐ It is necessary that the learning rate α (weight given to new samples) is decreased to 0 over time.
- ☐ It is necessary that the discount γ is less than 0.5.
- ☐ It is necessary that actions get chosen according to $\arg\max_a Q(s, a)$.

Question 6: Q-Learning Properties

0.0/4.0 points (graded)

In general, for Q-Learning to converge to the optimal Q-values...

- ☐ It is necessary that every state-action pair is visited infinitely often. ✓
- ☐ It is necessary that the learning rate α (weight given to new samples) is decreased to 0 over time. ✓
- ☐ It is necessary that the discount γ is less than 0.5.
- ☐ It is necessary that actions get chosen according to $\arg\max_a Q(s, a)$.

a) In order to ensure convergence in general for Q learning, this has to be true. In practice, we generally care about the policy, which converges well before the values do, so it is not necessary to run it infinitely often.

b) In order to ensure convergence in general for Q learning, this has to be true.

c) The discount factor must be greater than 0 and less than 1, not 0.5.

d) This would actually do rather poorly, because it is purely exploiting based on the Q-values learned thus far, and not exploring other states to try and find a better policy.