15-281 Artificial Intelligence HW5

INSTRUCTIONS

- **Due: Tuesday, October 10, 2023 at 10:00 PM EDT.** Remember that you may use up to 1 slip day for the Written Homework making the last day to submit **Wednesday, October 11, 2023 at 10:00 PM EDT**.
- **Format:** Write your answers in the `yoursolution.tex` file and compile a pdf (preferred) or you can type directly on the blank pdf. Make sure that your answers are within the dedicated regions for each question/part. If you do not follow this format, we may deduct points. Handwritten solutions are not acceptable and may lead to lost points.
- **How to submit:** Submit a pdf with your answers on Gradescope. Log in and click on our class 15-281, click on the appropriate assignment, and upload your pdf containing your answers.
- **Policy:** See the course website for homework policies and academic integrity.

Name	
Andrew ID	
Hours to complete?	◯ (0, 2] hours ◯ (2, 4] hours ◯ (4, 6] hours ◯ (6, 8] hours ◯ > 8 hours

Q1. [18 pts] Entailment

Consider the following Sudoku board:

	1		2
2		1	3
1			4
4			

Recall the rules of Sudoku (specifically for this 4x4 board):

1. Every row must contain the numbers 1-4 without repetitions
2. Every column must contain the numbers 1-4 without repetitions
3. Every 2x2 block in a corner must contain the numbers 1-4 without repetitions

Based on the given Sudoku board, answer the following questions about entailment. The positions on the board are represented as (row, col). Note that position (0,0) is the top left position on the board, and position (3,3) is the bottom right position. The notation "1:(0,0)" means putting a 1 at position (0,0).

(a) [8 pts]

(i) [2 pts] Does the given board $\models$ 3:(0,0)? Explain your answer.

Answer:

(ii) [2 pts] Does the given board $\models$ 2:(2,1)? Explain your answer.

Answer:

(iii) [2 pts] Does the given board $\models$ 1:(3,2)? Explain your answer.

Answer:

(iv) [2 pts] Does the given board $\models$ 3:(3,2)? Explain your answer.

Answer:

(b) [6 pts] Give three queries for empty squares whose values are entailed by the given board, using the same notation as the previous part. Do not use any of the entailment queries from part a.

Answer:

(c) [4 pts] Is there a solution to this board that can be found only through entailment? Note that this means new queries may be feasible once you fill in other squares that were entailed. If so, what is the solution? If not, give an example of a square where no value is entailed by the final board.

Answer:

Q2. [12 pts] SAT

For the following sentences, determine whether it is satisfiable or unsatisfiable. If satisfiable, select the model such that the sentence is satisfied. **If T and F both are both valid assignments for a variable, assign the variable to T.** If the model is unsatisfiable, select "Unsatisfiable."

Showing your work in these questions is optional, but it is recommended to help us understand where any misconceptions may occur.

(a) [4 pts] $\neg[\neg(\neg Y \wedge Y) \implies (\neg X \wedge \mathbf{F})] \wedge [(X \Leftrightarrow Y) \wedge \neg(X \vee \neg Y)]$

- ◯ X = False, Y = False
- ◯ X = False, Y = True
- ◯ X = True, Y = False
- ◯ X = True, Y = True
- ◯ Unsatisfiable

Work:

(b) [4 pts] $\neg[X \vee \neg(X \wedge (Y \vee \mathbf{T}))] \Rightarrow \neg[Z \wedge (\neg Z \vee (\mathbf{T} \Rightarrow \mathbf{F}))]$

- ◯ X = False, Y = True, Z = False
- ◯ X = False, Y = False, Z = False
- ◯ X = True, Y = True, Z = False
- ◯ X = True, Y = False, Z = False
- ◯ X = False, Y = True, Z = True
- ◯ X = False, Y = False, Z = True
- ◯ X = True, Y = True, Z = True
- ◯ X = True, Y = False, Z = True
- ◯ Unsatisfiable

Work:

(c) [4 pts] $[\neg(X \vee \neg X) \vee Y] \wedge [Y \vee (Z \Leftrightarrow \neg Z)]$

- ◯ X = False, Y = True, Z = False
- ◯ X = False, Y = False, Z = False
- ◯ X = True, Y = True, Z = False
- ◯ X = True, Y = False, Z = False
- ◯ X = False, Y = True, Z = True
- ◯ X = False, Y = False, Z = True
- ◯ X = True, Y = True, Z = True
- ◯ X = True, Y = False, Z = True
- ◯ Unsatisfiable

Work:

Q3. [10 pts] Resolution

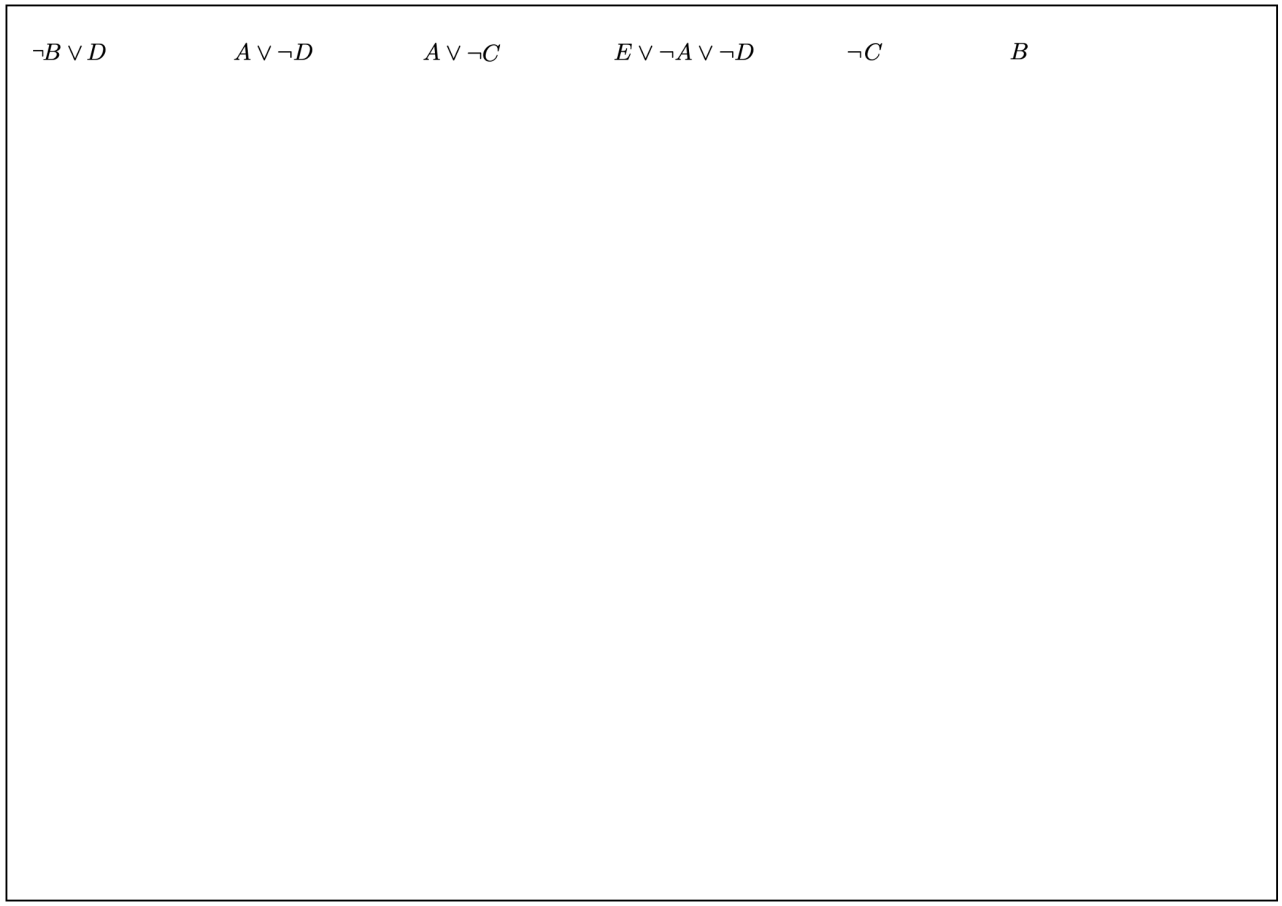
(a) [10 pts] Given the following propositional logic clauses, show E must be true by adding $\neg E$ and using only the resolution inference rule to derive a contradiction. Your answer should be in the form of a graph, where each resolvent is connected by lines to its two parent clauses. Use the clauses below as the initial set of nodes in the graph.

Note: You do not need to use all the nodes, and you may use a node more than once.

For your submission to this problem, you may do one of the following:

- Draw/annotate on top of the existing images in the pdf.
- Edit the `figures/resolution.png` image file to add markings.

Hand drawing is acceptable, as long as it is clear and precise enough.

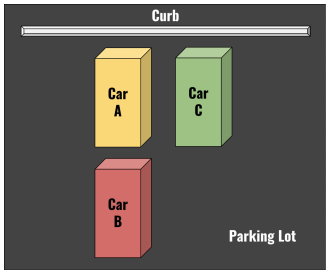


Q4. [16 pts] Valet Parking - Successor State Axioms

Suppose Alex has secured a side hustle as a valet driver for Pinky's exclusive restaurant, *Pinky's Deep Learning Dining*™, and his job is to park customer's cars in a lot while they dine.

The rules for parking and moving around cars in the lot are as follows:

- When Alex parks a car, he either parks in a new column, or parks behind another car.
- Alex can only occupy one car at a time to either park or move it.
- Alex can only move a car if there is no car behind it (so he can safely back out without hitting another vehicle).
- The curb is infinite so he can always park a car there.



Suppose there are symbols:

- ParkedBehind_[car]_[location]: car is behind location, where location is either a car or the curb.
- ClearBehind_[car]: the space behind car is clear.

and actions:

- ParkInNewCol_[car]: parks a car behind the curb so that it creates a new column with only itself in it.
- ParkBehindCar_[car1]_[car2]: parks car1 directly behind car2 in an existing column.

Consider the figure above as the state of the world at time $t = 1$.

(a) [2 pts] What is the state of the world (the KB) at time $t = 1$? Express your answer as a CNF sentence.

State at $t = 1$:

(b) [4 pts] What are ***all*** the possible states of the world at time $t = 0$? Express these states as CNF sentences. For each state, specify which action at time $t = 0$ leads to the state of the world at time $t = 1$ described above.

In this representation, we do not care about the left-to-right ordering of the cars. The only thing that matters is whether some cars are in the same column. **Note that "no-op" is NOT a valid action.**

Not all spaces provided may be necessary, if so write "N/A" in the blank spaces.

Possible State at $t = 0$:	**Action:**

Possible State at $t = 0$:	**Action:**

Possible State at $t = 0$:	**Action:**

Possible State at $t = 0$:	**Action:**

To flex his logical planning brain muscles, Alex attempts to write the successor state axioms for two fluents, **ParkedBehind_B_A** and **ClearBehind_B**, at the generic time $t+1$. He remembers the formula for a fluent F from class:

$$F^{t+1} \Leftrightarrow [F^t \wedge \neg ActionCausesNotF^t] \vee [\neg F^t \wedge ActionCausesF^t]$$

Since Alex wants to consider a generic time, $t+1$, he cannot just consider actions from the states described in parts **(a)** and **(b)**. He needs to consider transitions from ***any state***.

(c) [4 pts] Alex tries using the formula but is stuck. Help him complete the first successor state axiom for the predicate $ParkedBehind\_B\_A$ by filling in one predicate (or its negation) in each box. Some of the predicates are already filled in for you. Make sure to include superscripts denoting the time. You might not need all boxes. If you believe a box is unnecessary, leave it blank.

$ParkedBehind\_B\_A^{t+1} \Leftrightarrow$

$$\Big(\boxed{} \wedge \boxed{} \wedge \neg ParkInNewCol\_B^t \Big)$$
$$\bigvee \Big(\boxed{} \wedge \boxed{} \wedge \boxed{} \Big)$$
$$\bigvee \Big(\boxed{} \wedge \boxed{} \wedge \boxed{} \Big)$$

(d) [6 pts] Thanks to your help in the previous part, Alex was able to figure out more of the next successor state axiom for $ClearBehind\_B$. Help him complete the second successor state axiom. Again, fill each box with a predicate or its negation. Some of the predicates are already filled in for you. Make sure to include superscripts denoting the time. You might not need all boxes. If you believe a box is unnecessary, leave it blank.

$ClearBehind\_B^{t+1} \Leftrightarrow$

$$\Big(ClearBehind\_B^t \wedge \boxed{} \wedge \boxed{} \Big)$$
$$\bigvee \Big(\neg ClearBehind\_B^t \wedge ParkedBehind\_A\_B^t \wedge \boxed{} \Big)$$
$$\bigvee \Big(\neg ClearBehind\_B^t \wedge \boxed{} \wedge ParkInNewCol\_A^t \Big)$$
$$\bigvee \Big(\neg ClearBehind\_B^t \wedge \boxed{} \wedge \boxed{} \Big)$$
$$\bigvee \Big(\neg ClearBehind\_B^t \wedge \boxed{} \wedge \boxed{} \Big)$$

Q5. [24 pts] Classical Planning and GraphPlan

Suppose we translate the Valet Parking problem into a classical planning problem with predicates ClearBehind(car) and ParkedBehind(car1,car2). We define two operations:

ParkBehind(car1, car2)

- Preconditions:
  - ClearBehind(car1)
  - ClearBehind(car2)
  - ParkedBehind(car1, place)
- Add List:
  - ParkedBehind(car1, car2)
  - ClearBehind(place)
  - ¬ ParkedBehind(car1, place)
  - ¬ ClearBehind(car2)
- Delete List:
  - ClearBehind(car2)
  - ParkedBehind(car1, place)

ParkInNewRow(car)

- Preconditions:
  - ClearBehind(car)
  - ParkedBehind(car, place)
  - ¬ ParkedBehind(car, curb)
- Add List:
  - ParkedBehind(car, curb)
  - ClearBehind(place)
  - ¬ ParkedBehind(car, place)
- Delete List:
  - ParkedBehind(car, place)
  - ¬ ParkedBehind(car, curb)

We have discussed two types of planning: **linear** and **non-linear planning**. Linear planning works on one goal until it is completely solved before moving on to the next goal. In contrast, non-linear planning considers all possible sub-goal orderings and handles goal interactions by interleaving. The issue with non-interleaved planning methods such as linear planning is that it will naively pursue one subgoal X after satisfying another subgoal Y, but may perform extra steps or may never accomplish the goal because steps required to accomplish X might undo things in subgoal Y. This issue has been coined the Sussman anomaly.

(a) [8 pts] With the following initial state, identify the solution plans a linear and non-linear planner would return using the operators above. Both linear and nonlinear planners will try goals from left to right.

$$State = ParkedBehind(C, A) \wedge ParkedBehind(A, Curb) \wedge$$

$$ParkedBehind(B, Curb) \wedge ClearBehind(B) \wedge ClearBehind(C)$$

NOTE: This state does not correspond to the diagram presented in question 4.
Assume all appropriate negated predicates are also in the knowledge base.

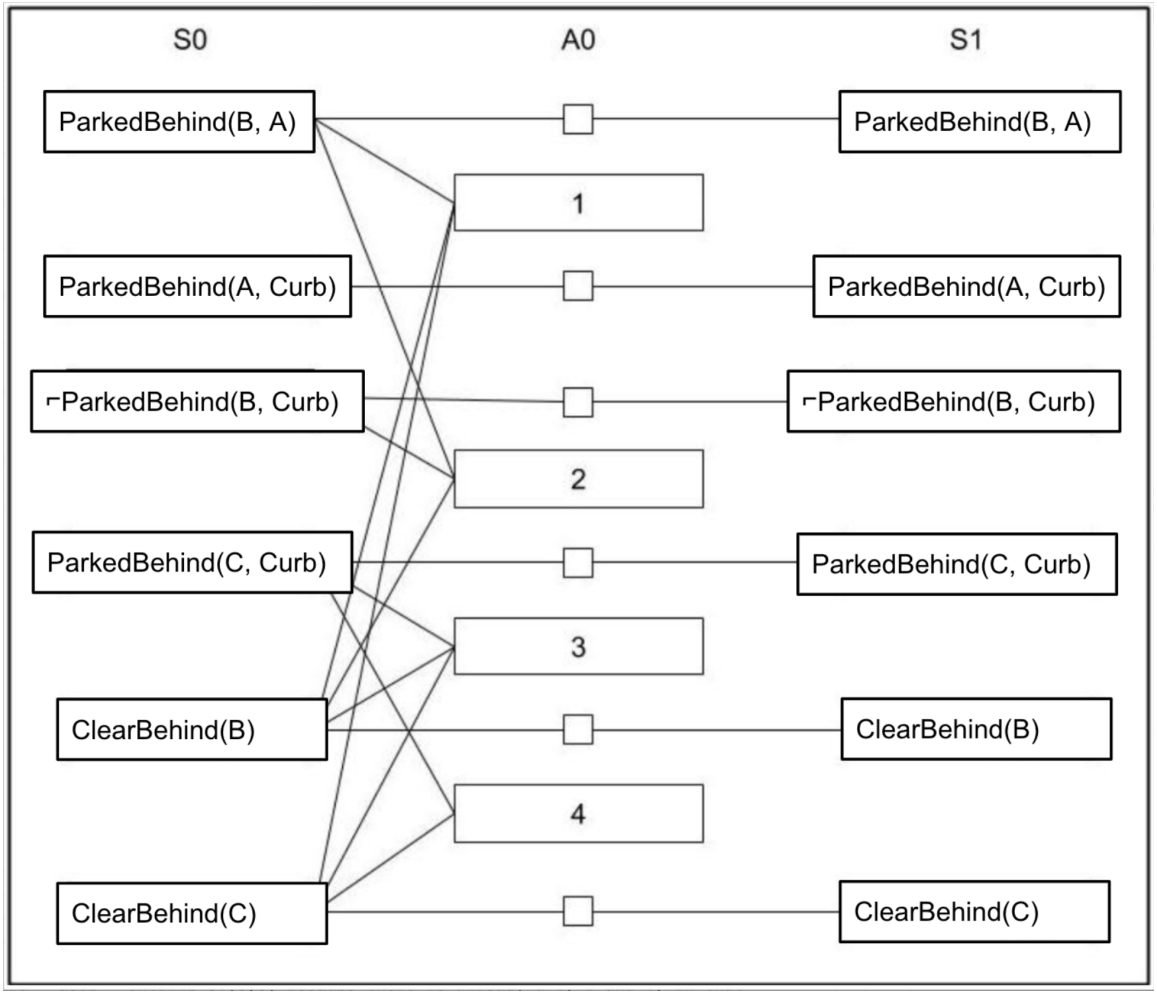
$$Goal = ParkedBehind(A, B) \wedge ParkedBehind(B, C) \wedge ParkedBehind(C, Curb)$$

Linear plan:

Non-linear plan:

(b) [4 pts] Now consider the following image that shows a template for the first two levels of the **GraphPlan graph** for a ValetParking problem. We have drawn in the connections between actions in A0 and their preconditions in S0, as well as persistence actions (unnamed action nodes or **no-ops**). Your task is to:

- Fill in the blanks for the appropriate action nodes in A0 for the boxes labeled 1-4 below.
- Write "N/A" if there is no possible action for the given preconditions. NOTE: normally, when running GraphPlan we won't include such N/A boxes.



1:	**2:**	**3:**	**4:**

(c) [4 pts] Which edges are connected to the state layer S1 as a result of each of the above actions?

- List all the nodes (predicates) in S1 to which there is an ***add*** edge from each of the following actions
- Write "N/A" if the action was not possible
- NOTE: not all predicate nodes are shown in S1 above but you should still include ALL relevant predicates in your response.

1:	**2:**
3:	**4:**

For the following questions, remember that no-op actions count as actions. If you want to use these actions, refer to them as No-op(state) where the precondition and result of No-op(state) is the "state" predicate.

(d) [2 pts] In your completed GraphPlan graph, name two action nodes between which there is an *Inconsistent effects* mutex relation.

Node 1:	**Node 2:**

(e) [2 pts] In your completed GraphPlan graph, name two action nodes between which there is an *Interference* mutex relation.

Node 1:	**Node 2:**

(f) [4 pts] One of the conditions for the GraphPlan algorithm to terminate with a failure is that the graph has **leveled off**. What does this mean? (Choose only one answer)

- ◯ A) All possible actions have been explored.
- ◯ B) There is no non-empty set of literals between which there are no mutex links.
- ◯ C) Two consecutive levels are identical.
- ◯ D) The last level of states contains a goal state.

Q6. [20 pts] Planning

Consider a planning environment with six different operations (defined in the table below), starting state A, and goal condition $C \wedge D \wedge E$. Only one operation may be applied at a time, and we are trying to find the plan with the fewest number of operations.

	op1	**op2**	**op3**	**op4**	**op5**	**op6**
Precondition	A	B	A	A	A	A
Add	B	C, D, E	C	D	E	E, ¬A
Delete						

(a) [5 pts]

(i) [3 pts] Run linear planning on this environment with the order of subgoals: C then D then E. What plan is returned?

Plan:

(ii) [1 pt] Is that plan optimal?

◯ Yes ◯ No

(iii) [1 pt] Explain your answer to part (ii).

Answer:

(b) [15 pts]

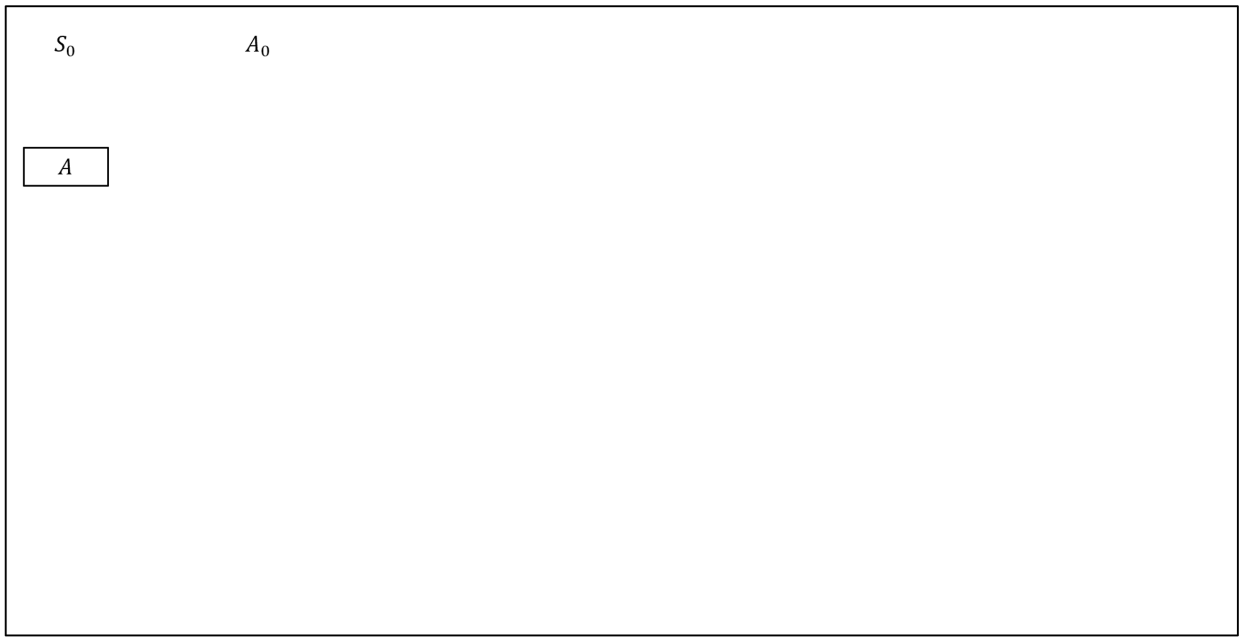
(i) [4 pts] Run GraphPlan on this environment. Draw the **GraphPlan graph**, adding action levels and proposition levels until GraphPlan terminates.

Note: make sure to include the No-op actions for persistent states in your drawing.

For your submission to this problem, you may do one of the following:

- Draw/annotate on top of the existing images in the pdf.
- Edit the `figures/graphplan.png` image file to add markings.

Hand drawing is acceptable, as long as it is clear and precise enough.


S_0
A_0
A

(ii) [3 pts] What plan is returned by GraphPlan?

Plan:

(iii) [2 pts] Is that plan optimal?

◯ Yes ◯ No

(iv) [6 pts] List ALL pairs of exclusive operators in A_0 and ALL pairs of exclusive propositions in S_1. Write 'None' if none exist.

Note: Remember that no-op counts as an action.

Exclusive Operators in A_0:	**Exclusive Propositions in S_1:**