Last Name: First Name:

Matriculation Number:

Retake Exam
Artificial Intelligence 1

October 9, 2023

Please ignore the QR codes; do not write on them, they are for grading support

	To be used for grading, do not write here													
prob.	1.1	2.1	2.2	3.1	4.1	4.2	5.1	5.2	5.3	6.1	7.1	7.2	Sum	grade
total	8	10	9	6	7	7	6	6	7	8	10	6	90	
reached														

The "solutions" to the exam/assignment problems in this document are supplied to give students a starting point for answering questions. While we are striving for helpful "solutions", they can be incomplete and can even contain errors even after our best efforts.
In any case, grading student's answers is not a process of simply "comparing with the reference solution", therefore errors in the "solutions" are not a problem in this case. If you find "solutions" you do not understand or you find incorrect, discuss this on the course forum and/or with your TA and/notify the instructors. We will – if needed – correct them ASAP.

In the course Artificial Intelligence I/II we award bonus points for the first student who reports a factual error in an old exam. (Please report spelling/formatting errors as well.)

1 Prolog

Problem 1.1 (Prolog in Prolog)
Consider the following Prolog program that represents Prolog in Prolog, i.e. Prolog terms, literals, and clauses are represented as Prolog terms:

```
1  isTerm(pterm(F,ARGS)) :- string(F), isTermList(ARGS).
2  isTerm(pvar(X)) :- string(X).
3
4  isTermList([]).
5  isTermList([H|T]) :- isTerm(H), isTermList(T).
6
7  isLiteral(plit(P,ARGS)) :- string(P), isTermList(ARGS).
8
9  isLiteralList([]).
10 isLiteralList([H|T]) :- isLiteral(H), isLiteralList(T).
11
12 isClause(pclause(H,B)) :- isLiteral(H), isLiteralList(B).
```

Here `string` is a built-in predicate that succeeds if its argument is a string.

1. Write the Prolog clause `isNat(succ(N)) :- isNat(N)` as a Prolog term relative to the above program (i.e., such that `isClause` succeeds for it). 3 Points

   Solution:

   ```
   pclause(plit("isNat", [pterm("succ",[pvar("N")])]),
           [plit("isNat", [pvar("N")])]).
   ```

2. Assume that the Prolog term C contains no free variables. How is the result of the query `isClause(`C`)` affected by exchanging the lines 4 and 5? 2 Points

   Solution: It is not affected.

3. Extend the program above with a unary Prolog predicate `isProgram` that succeeds if its argument is of the form `pprog(`P`)` where P is a list of clauses. 3 Points

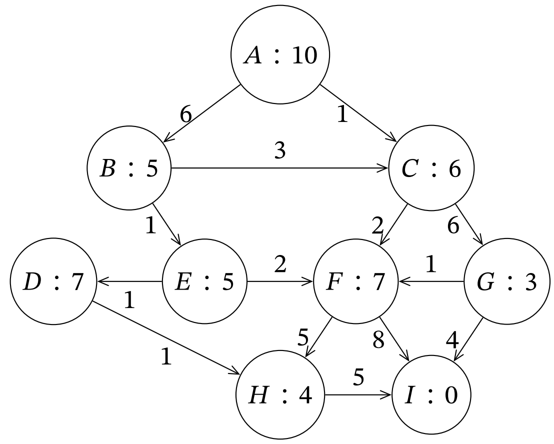
   Solution:

   ```
   isClauseList([]).
   isClauseList([H|T]) :- isClause(H), isClauseList(T).

   isProgram(pprog(C)) :- isClauseList(C).
   ```

2 Search

Problem 2.1 (Search Algorithms)
Consider the following directed graph:



Every node is labeled with $n : h(n)$ where n is the identifier of the node and $h(n)$ is the heuristic for estimating the cost from n to a goal node. Every edge is labeled with its actual cost.

1. Assume that I is the goal node. Argue whether or not the heuristic is admissible. 2 Points

Solution: It is not admissible: The cost from D to the goal is $1 + 5 = 6 < 7 = h(D)$, and a heuristic must not overestimate that cost.

Now assume you have already expanded the node A. List the **next** 4 **nodes (i.e., excluding** A**)** that will be expanded using the respective algorithm. If there is a tie, break it using alphabetical order.

2. depth-first search 1 Points

Solution: B, C, F, H

3. breadth-first search 1 Points

Solution: B, C, E, F

4. uniform-cost search 2 Points

Solution: C, F, B, E

5. greedy-search 2 Points

Solution: B, E, C, G

6. A^*-search 2 Points

Solution: C, F, G, B

Problem 2.2 (Search Problems)
Consider the search problem $\langle S, A, T, I, G\rangle$ where

- ☐ $S = \mathbb{Z} \times \mathbb{Z}$
- ☐ $A = \{R, S, M\}$
- ☐ T is given by
  - ☐ $T(R,(x,y)) = \{(x,0),(0,y)\}$
  - ☐ $T(S,(x,y)) = \{(y,x)\}$
  - ☐ $T(M,(x,y)) = \{(x+1,y)\}$
- ☐ $I = \{(0,0)\}$
- ☐ $G = \{(3,3)\}$

1. Tick the box of the part of the definition that makes this problem fully observable. 1 Points

Solution: the one for I

2. Give the possible states resulting from applying the action sequence M, R, M to the initial state. 2 Points

Solution: $(2,0)$, $(1,0)$

3. Which states are reachable from the initial state? 2 Points

Solution: all states with non-negative coordinates

4. Give a solution of minimal length to this problem. 2 Points

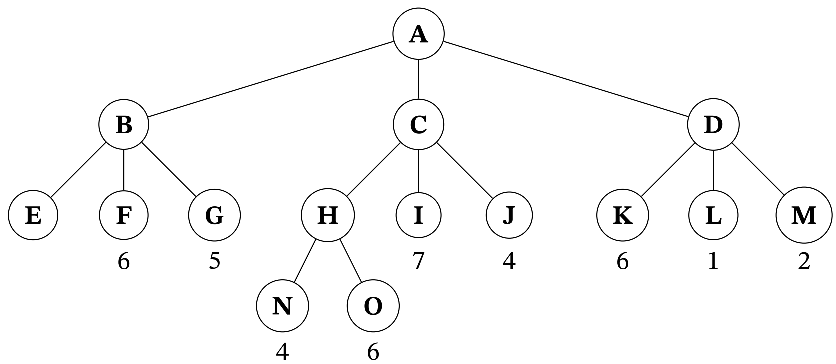
Solution: M, M, M, S, M, M, M

5. Assume we use $h((x,y)) = 1/(1+x+y)$ as a heuristic. Whichs action will a greedy search algorithm choose for the first two steps? 2 Points

Solution: M, M

3 Adversarial Search

Problem 3.1 (Minimax)
Consider the following minimax game tree for the **maximizing** player's turn. The values at the leafs are the static evaluation function values of those states; some of those values are currently missing.



1. Label the nodes H and C with their minimax values. 2 Points

   Solution: H: 6, C: 4

2. If possible, label the node **E** with an evaluation function value that results in the player definitely choosing move **C** (no matter how ties are broken).
   Otherwise, argue why that is impossible. 2 Points

   Solution: Any label < 4.

3. Now assume **E** is labeled with 5, and we use $\alpha\beta$-pruning. We expand child nodes in alphabetical order. Which nodes would be pruned? 2 Points

   Solution: M

4 Constraint Satisfaction/Propagation

Problem 4.1 (Modeling) 7 Points

You want to schedule a tournament in which teams A, B, C, D play each other once. The six games must take place over the next 3 days. But team A must not play twice on the same day. Team B is only available for the next 2 days. And team C wants to play against D a day before playing against anybody else.

Model this problem as a constraint satisfaction problem $\langle V, D, C\rangle$. Explain how the solutions correspond to the possible match schedules.

Solution: E.g.,
$V = \{AB, AC, AD, BC, BD, CD\}$
$D_{XY} = \{1, 2, 3\}$ for all $XY \in V$
Constraints in C:

- A-matches on different days: $AB \neq AC$, $AB \neq AD$, $AC \neq AD$
- B-matches on first two days: $AB \leq 2$, $BC \leq 2$, $BD \leq 2$

- CD-match before other C-matches: $CD < AC$, $CD < BC$

Explanation: For a solution s, the match XY is played on day $s(XY)$.

Problem 4.2 (Solving)
Consider the following binary CSP:

- $V = \{a, b, c, d, e\}$
- $D_a = D_b = D_c = \{0, 1, 2, 3\}$, $D_d = D_e = \{0, 1, 2, 3, 4, 5, 6, 7\}$
- Constraints:
  - $e^2 - d^2 < 18$
  - $d = 2c$
  - $e - a > 5$
  - $a < b$ and $b < c$ and $d < e$

1. Check the boxes for (v, w) if v is arc-consistent relative to w. 2 Points

   ☐ (a, b) ☐ (b, a) ☐ (a, e) ☐ (e, a) ☐ (c, d) ☐ (d, c)

   Solution: Only (c, d).

2. Give the **three** solutions. 3 Points

   Solution: $(a, b) \in \{(0, 1), (0, 2), (1, 2)\}$, $c = 3$, $d = 6$, $e = 7$

3. Now assume we replace the last constraint with $b < \min\{c, d\}$ (where min is the minimum operator). Transform the resulting problem into an equivalent binary one. 2 Points

   Solution: We can replace the new constraint with the constraints $b < c$ and $b < d$. In fact, the constraint $b < c$ suffices because then $b < d$ can be inferred from the existing constraints.

5 Logic

Problem 5.1 (Propositional Logic)
We use the propositional variables X, Y, and Z. Consider the formula A given by

$$(X \wedge (Y \Rightarrow Z)) \Rightarrow \neg(X \wedge Y)$$

1. Give a satisfying assignment σ and a falsifying assignment φ for A. 3 Points

   Solution: The satisfying and falsifying assignments can be read off the following table. $\sigma(X) = \sigma(Y) = \sigma(Z) = 1$ and φ any other assignment.

X	Y	Z	interpretation of A
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

2. Which (if any) of the formulas A and $\neg A$ is a theorem? 1 Points

Solution: Neither (because A can be both satisfied and falsified).

3. Give the shortest formula in CNF that is equivalent to $A \Rightarrow A$. 2 Points

Solution: $true$ (because the formula is a theorem for any value of A)

Problem 5.2 (Predicate Logic)

Consider the following signature of predicate logic:

- binary function symbol f
- unary predicate symbol p

1. Give a model for that signature. 2 Points

Solution: E.g., universe $\mathbb{N}$, $\mathcal{I}(f)(u,v) = u + v$, $\mathcal{I}(p) = \{0\}$.

2. Consider the formula $A = p(x) \wedge p(y)$ and assume a model with universe $\mathbb{N}$ and $\mathcal{I}(p) = \{0\}$. Give an assignment α such that $\mathcal{I}_\alpha(A)$ holds. 2 Points

Solution: $\alpha(x) = \alpha(y) = 0$

3. Prove or refute the following statement: A model that satisfies $\forall x.p(x)$ satisfies all formulas. 2 Points

Solution: False. E.g., $false$ is a counter-example.

Problem 5.3 (Proving in Tableau Calculus) 7 Points

We use the propositional variables P, Q, and R and the following abbreviations

$$A = Q \wedge (P \Rightarrow Q)$$

$$B = P \Rightarrow A$$

$$C = P \Rightarrow R$$

Using the tableau calculus, find a falsifying assignment for the formula $C \Rightarrow B$.

Solution:

$$\begin{array}{c}
C \Rightarrow B^F \\
\hline
C^T \\
B^F \\
P^T \\
A^F \\
\begin{array}{c|c}
P^F & R^T \\
\text{close on } P & \begin{array}{c|c} Q^F & P \Rightarrow Q^F \\ \text{open: } P^T, Q^F, R^T & P^T \\ & Q^F \\ & \text{open: } P^T, Q^F, R^T \end{array}
\end{array}
\end{array}$$

So the only falsifying assignment φ is given by $\varphi(P) = \varphi(R) = 1$ and $\varphi(Q) = 0$

---

6 Knowledge Representation

Problem 6.1 (Description Logic)
Consider the following $\mathcal{ALC}$-setting:

- concepts: `pizza`, `icecream`, `food`, `topping`
- relations: `canHaveTopping`
- individuals: `margarita`, `vanilla`, `ham`, `syrup`

You may abbreviate every concept/relation/individual by its first letter.

1. Give the $\mathcal{ALC}$-ABox with assertions that model common sense knowledge (e.g., we do not put vanilla or syrup on pizza even though it is technically possible). 2 Points

---

Solution: $m : p, v : i, h : t, s : t, p\ c\ h, i\ c\ s$

---

2. Give an $\mathcal{ALC}$-TBox with two axioms expressing the following: 2 Points
   - All food is pizza or icecream.
   - Only pizza can have toppings.

Solution: $f \sqsubseteq p \sqcup i, \exists c.t \sqsubseteq p$

---

3. Give an $\mathcal{ALC}$-TBox in which the concept `pizza` is inconsistent. 2 Points

---

Solution: E.g., $p \sqsubseteq \bot$

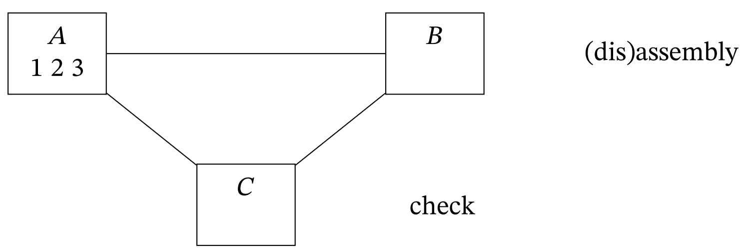
---

4. Give the result of translating the following formula to first-order logic: $(\forall c.i) \sqsubseteq (p \sqcap t)$ 2 Points

Solution: $\forall x.(\forall y.c(x,y) \Rightarrow i(y)) \Rightarrow p(x) \wedge t(x)$

7 Planning

Problem 7.1 (STRIPS)
Consider a machine that processes objects $Obj = \{1, 2, 3\}$, which can be at location A, B, or C. Currently all objects are at location A and **unchecked** and **assembled**. Eventually all objects are needed in location A and **checked** and **assembled**.
At location B, objects can be assembled or disassembled. At location C, disassembled objects can be checked.
A transport system is available that can move **exactly two objects at a time** from one location to any other location.



We formalize this problem as a STRIPS task where the **facts** are

- position $l \in \{A, B, C\}$ of object $o \in Obj$: $\texttt{at}(l, o)$
- state of object $o \in Obj$: $\texttt{isCh}(o)$ (checked), $\texttt{isAss}(o)$ (assembled), and $\texttt{isDis}(o)$ (disassembled)

and the **actions** are given by table below for any $l, m \in \{A, B, C\}$ and $o, p \in Obj$

action	precondition	add list	delete list
$\texttt{move}(l, m, o, p)$	$\texttt{at}(l, o)$, $\texttt{at}(l, p)$	$\texttt{at}(m, o)$, $\texttt{at}(m, p)$	$\texttt{at}(l, o)$, $\texttt{at}(l, p)$
$\texttt{assemble}(o)$	$\texttt{at}(B, o)$	$\texttt{isAss}(o)$	$\texttt{isDis}(o)$
$\texttt{disassemble}(o)$	$\texttt{at}(B, o)$	$\texttt{isDis}(o)$	$\texttt{isAss}(o)$
$\texttt{check}(o)$	$\texttt{at}(C, o)$, $\texttt{isDis}(o)$	$\texttt{isCh}(o)$	

1. Give the initial state I and the goal G. 2 Points

Solution: initial state: $\texttt{at}(A, o)$ and $\texttt{isAss}(o)$ for all $o \in Obj$,
goal: $\texttt{at}(A, o)$, $\texttt{isCh}(o)$ and $\texttt{isAss}(o)$ for all $o \in Obj$

2. Give the state after applying the two actions $\texttt{move}(A, B, 1, 2)$ followed by $\texttt{disassemble}(1)$. 2 Points

Solution: $\texttt{at}(A, 3)$, $\texttt{at}(B, 1)$, $\texttt{at}(B, 2)$, $\texttt{isDis}(1)$, $\texttt{isAss}(2)$, $\texttt{isAss}(3)$

3. Give the value $h^*(I)$. 2 Points

Solution: 17 (3 actions per object for disassemble, check, assemble, as well as 2 sequences of 4 move actions $A - B - C - B - A$ for pairs of objects)

4. Give the value $h^+(I)$. 2 Points

Solution: 10. (With the delete heuristic, objects do not have to re-assembled or moved back. So only 2 actions are needed per object instead of 3 and only 2 moves per pair of objects.)

5. Consider the heuristics h that computes $h(s)$ as $2a + d + 3$ where a is the number of unchecked assembled and d the number of unchecked disassembled objects in state s. Argue whether h is admissible. 2 Points

Solution: It is not admissible. A counter-example is the goal state g where $h^*(g) = 0 < 3 = h(g)$ in violation of the admissibility condition.

Problem 7.2 (Planning Complexity)

1. What is the difference between satisficing and optimal planning? 2 Points

Solution: Satisficing planning searches for any plan. Optimal planning for one with minimal length.

2. Give the named complexity class (e.g., P, NP, etc.) of deciding the existence of a plan for a STRIPS problem. 1 Points

Solution: PSPACE (which is the same as NPSPACE)

3. Now we consider only STRIPS problems in which all delete lists are empty. Show that the number of facts is an upper bound for the length of an optimal plan. 3 Points

Solution: Without delete lists, an action cannot decrease the number of facts in the state. Actions that do not change the state are redundant. So every action in an optimal plan increases the state. The number of facts is an upper bound for how often the state can be increased and thus for the length of an optimal plan.