Last Name: First Name:

Matriculation Number:

Seat:

Exam
Artificial Intelligence 1

August 1, 2022

	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	10	8	7	6	7	8	5	8	8	9	12	8	96	
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 5 bonus points for the first student who reports a factual error (please report spelling/-formatting errors as well) in an assignment or old exam and 10 bonus points for an alternative solution (formatted in LaTeX) that is usefully different from the existing ones.

1 Prolog

Problem 1.1 (Prolog) 10 pt

Consider the following Prolog code for propositional logic, where we represent the propositional variables as `pv(N)` for natural numbers `N`.

```
isNat(zero).
isNat(succ(N)) :- isNat(N).

isForm(conj(F,G)) :- isForm(F), isForm(G).
isForm(neg(F)) :- isForm(F).
isForm(pv(N)) :- isNat(N).
```

1. What do the following queries return: 3 pt

   (a) `isForm(conj(pv(zero),pv(zero))`

   (b) `isForm(X)`

   If a query returns multiple results, only give the first two.

2. Write a Prolog program `uses` such that `uses(F,N)` holds if the propositional variable with number `N` occurs in the formula F. 5 pt

3. Assume we change the `isForm` predicate to 2 pt

   ```
   isForm(conj(F,G)) :- isForm(G), isForm(F).
   ```

   Explain if and how this affects which results are returned by the query `isForm(conj(pv(M),pv(N)))`.

Solution:

1. (a) `True`
   (b) No results are returned because the query backtracks forever.

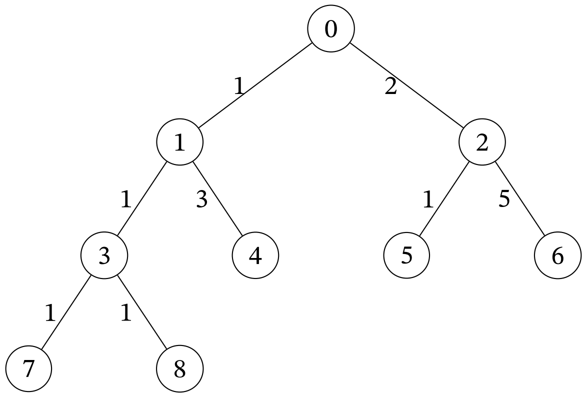
2.
```
isNat(zero).
isNat(succ(N)) :- isNat(N).

isForm(conj(F,G)) :- isForm(F), isForm(G).
isForm(neg(F)) :- isForm(F).
isForm(pv(N)) :- isNat(N).

uses(conj(F,_), N) :- uses(F,N).
uses(conj(_,G), N) :- uses(G,N).
uses(neg(F), N) :- uses(F,N).
uses(pv(N), N).
```

3. It will find different formulas, namely $p_n \wedge p_0$ instead of $p_0 \wedge p_n$ for increasing n.

2 Search

Problem 2.1 (Search Algorithms) 8 pt

Consider the following tree with root 0:



Every edge is labeled with its cost.
As a heuristic for estimating the distance from node n to a goal node, we use $h(n) = 8 - n$.

Assume you have already expanded the root node. List the **next** 4 nodes that will be expanded using

1. depth-first search 1 pt

2. breadth-first search 1 pt
3. uniform-cost search 2 pt
4. greedy search 2 pt
5. A^*-search 2 pt

If there is a tie, first expand the node with the smaller number.

Solution:

1. $1, 3, 7, 8$
2. $1, 2, 3, 4$
3. $1, 2, 3, 5$
4. $2, 6, 5, 1$
5. $1, 3, 8, 7$

Problem 2.2 (Search Problems) 7 pt

Consider the search problem (S, A, T, I, G) where

- $S = \mathbb{Z}$
- $A = \{-2, -1, 0, 1, 2\}$
- $T(a, s) = \{a \cdot s\}$
- $I = \{1\}$
- $G = \{8\}$

1. Give the state resulting from applying the action sequence $-1, 2, -2$ to the initial state. 1 pt
2. Give a solution to the problem. 3 pt
3. Assume we try actions in the order $-2, -1, 0, 1, 2$. Will DFS find a solution? Why (not)? 2 pt
4. Give all states that are reachable from the initial state. 1 pt

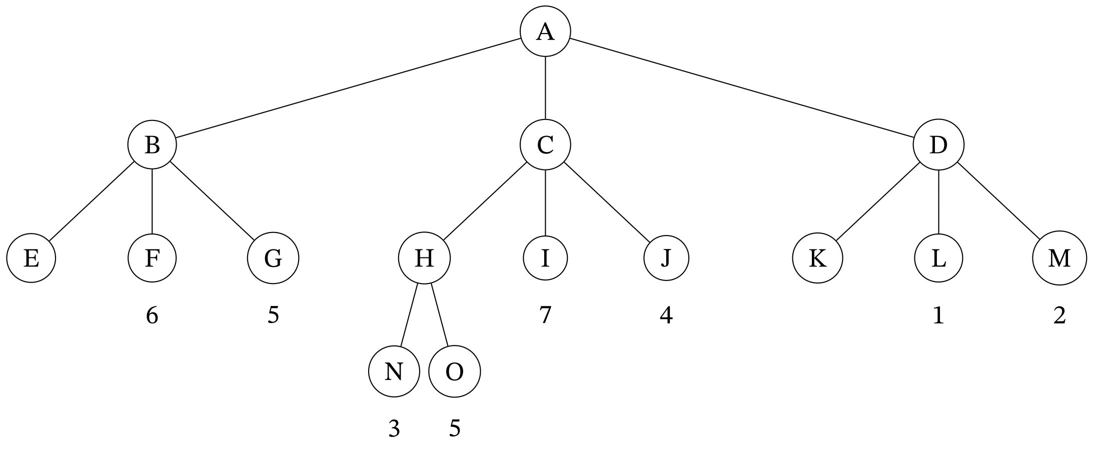
Solution:

1. 4
2. 2, 2, 2 (any sequence of actions that multiply to 8)
3. No. It will only try -2 and find the states $1, -2, 4, -8, 16, \ldots$.
4. All powers of 2 and their negations and 0.

3 Adversarial Search

Problem 3.1 (Minimax) 6 pt

Consider the following minimax game tree (**without** alpha-beta pruning) for the **minimizing 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 pt
2. Label the nodes E and K with evaluation function values that result in the player choosing move B. 2 pt
3. Now assume E is labeled 1 and K is labeled 8.
   Which out of K, L, M are α-β-search pruned (i.e., not visited)?
   Assume children of a node are visited in alphabetical order. 2 pt

Solution:

1. H= 3, C= 7
2. E $\leq 7, K \geq 7$.
3. L and M

4 Constraint Satisfaction/Propagation

Problem 4.1 (Pieces on a 3D Board) 7 pt

Consider a cubic board of size $3 \times 3 \times 3$ (i.e., there are 27 spots to place a piece). We want to place 2 pieces in such a way that no pieces touch via a face in any direction (up/down, left/right, forward/backward). Touches via an edge or a corner are allowed.

1. Model the problem as a constraint satisfaction problem (V, D, C). 6 pt
2. Explain your model briefly by saying how piece placements correspond to the assignments for the problem. 1 pt

Make sure you give a formally exact definition, i.e., explicitly define the sets V and all sets D_v. You can describe each constraint as a set of tuples or as a formula.

Solution:

1. $V = a_1, a_2, a_3, b_1, b_2, b_3$
   $D_v = \{0, 1, 2\}$ for all $v \in V$
   Constraints in C:
   - $|a_1 - b_1| + |a_2 - b_2| + |a_3 - b_3| > 1$
2. The assignments to (a_1, a_2, a_3) and (b_1, b_2, b_3) correspond to the coordinates of the two spots where the pieces are placed.

Problem 4.2 (Solving) 8 pt

Consider the following binary CSP:

- $V = \{a, b, c\}$
- $D_a = \{0, 1, 2\}, D_b = \{0, 1, 2, 3\}, D_c = \{0, 1, 2, 3, 4\}$
- Constraints:
  - if $a > b$, then $b = 2$
  - if $a < c$, then $c \neq 2$
  - $b + c < 4$ and $b < c$

1. Give all pairs (v, w) of variables such that v is **not** arc-consistent relative to w. 3 pt
2. Give all solutions. 3 pt
3. Assume we assign a to be 1, and apply forward-checking. Give the resulting domains D_b and D_c. 2 pt

---

Solution:

- (b, c), (c, b)
- There are 2 solutions: $a = 0$, $b = 0$, $c \in \{1, 3\}$.
- $D_b = \{1, 2, 3\}$, $D_c = \{0, 1, 3, 4\}$

---

5 Logic

Problem 5.1 (Propositional Logic) 5 pt
We use the propositional variables P, Q, R. Consider the formula A given by

$$(P \vee Q) \Rightarrow ((R \vee P) \wedge (R \Rightarrow Q))$$

1. Using the assignment $\varphi(P) = F$, $\varphi(Q) = T$, and $\varphi(R) = T$, give the value $I_\varphi(A)$. 2 pt
2. Argue whether A is valid (i.e., give a proof or a counter-example). 3 pt

---

Solution:

1. T
2. It is not valid. For example, the assignment with $\varphi(P) = \varphi(R) = T$ and $\varphi(Q) = F$ falsifies it and is thus a counter-example.

Problem 5.2 (Definitions) 8 pt
Consider the following signature of first-order logic:

- binary function symbol f
- unary predicate symbol P

Give counter-examples for the following statements:

1. For every satisfiable formula A, the formula $\neg A$ is unsatisfiable. 2 pt

2. Every model that satisfies $\forall x.P(f(x,x))$ also satisfies $\forall x.\forall y.P(f(x,y))$. 3 pt

3. Consider the model with universe $\mathbb{Z}$, $I(P) = \{n \in \mathbb{Z} | n > 5\}$, and $I(f)(m,n) = m - n$. Then for every assignment φ, we have 3 pt

$$I_\varphi\big(P(f(x,y)) \Rightarrow P(f(y,x))\big) \;=\; T.$$

Solution:

1. Any non-theorem non-contradiction A, e.g., $A = \forall x.P(x)$
2. E.g., $I(P) = \mathbb{N}$, $I(f) = +$, $I(P) = even$.
3. Any assignment with $\varphi(x) - \varphi(y) > 5$.

Problem 5.3 (Proving) 8 pt
Prove the following formula using tableaux calculus:

$$(\forall x.P(x) \Rightarrow \neg Q(x)) \Rightarrow \big((\exists y.P(y)) \Rightarrow \neg\forall z.Q(z)\big)$$

Solution:

$$\begin{array}{c}
(\forall x.P(x) \Rightarrow \neg Q(x)) \Rightarrow \big((\exists y.P(y)) \Rightarrow \neg\forall z.Q(z)\big)^F \\
\hline
(\forall x.P(x) \Rightarrow \neg Q(x))^T \text{ (1)} \\
(\exists y.P(y)) \Rightarrow \neg\forall z.Q(z)^F \\
(\exists y.P(y))^T \text{ (2)} \\
\neg\forall z.Q(z)^F \\
\forall z.Q(z)^T \text{ (3)} \\
P(s)^T \text{ (from (2) for fresh } s) \\
Q(s)^T \text{ (from (3) with } s) \\
P(s) \Rightarrow \neg Q(s)^T \text{ (from (1) with } s) \\
\begin{array}{c|c}
P(s)^F & \neg Q(s)^T \\
 & Q(s)^F \\
\text{close via } P(s) & \text{close via } Q(s)
\end{array}
\end{array}$$

---

6 Knowledge Representation

Problem 6.1 (Specifying Properties in ALC) 9 pt
Consider the following ALC setting:

- concepts: `human`, `child`, `grownup`, `animal`
- relations: `isChildOf`, `owns`

We abbreviate every concept/relation by its first letter.

1. Give an ALC ABox that is not consistent with the axiom $h \sqcap \exists o.a = \bot$. 2 pt
2. Give an ALC TBox that formalizes the following properties 3 pt
   - Children and grownups are humans, and humans are children or grownups. No one is both a child and a grownup.
   - Humans cannot be owned.
3. Give an ALC formalization for the concept of children who have a parent who owns an animal. 2 pt
4. Give the translation to first-order logic of the ALC statement $(\forall i.h) \sqsubseteq (\exists i.g)$. 2 pt

---

Solution:

1. $x : h, y : a, x\, o\, y$
2. We use the axioms $h = c \sqcup g$, $c \sqcap g = \bot$, and $\exists o.h = \bot$
3. $c \sqcap \exists i.\exists o.a$
4. $\forall x.\big(\forall y.i(x,y) \Rightarrow h(y)\big) \Rightarrow \big(\exists y.i(x,y) \wedge g(y)\big)$

---

7 Planning

Problem 7.1 (STRIPS) 12 pt

Consider a STRIPS task $\mathcal{T} = \langle P, A, I, G\rangle$. Recall that the actions $a \in A$ are triples (pre_a, add_a, del_a).

1. Explain the meaning of pre_a, add_a, and del_a. 4 pt
2. What happens if $G \subseteq I$? 2 pt
3. What is the definition and motivation of the delete relaxation? 3 pt
4. We can define a deterministic search problem $\langle S', A', T', I', G'\rangle$ where 3 pt
   - S' is the power set of P
   - $I' = I$
   - the solutions are exactly the plans for $\mathcal{T}$.

   Give

   (a) the set $A'(s)$ of actions applicable in state s:
   (b) the state $T'(a, s)$ resulting from applying action a in state s:
   (c) the set G' of goal states:

Solution:

1. All three are sets of facts. Those in pre_a must hold for a to applicable. After applying a, those in add_a hold and those in del_a do not.

2. The goal condition is already satisfied in the initial state. So the empty sequence of actions is a plan for $\mathcal{T}$.

3. It makes the sets del_a empty. Thus, facts never become false, thus preconditions are easier to satisfy, and optimal plans become shorter and easier to find. Therefore, it can be used as a heuristic.

4. $A'(s)$ is the set of actions $a \in A$ for which $pre_a \subseteq s$; $T(a,s) = s \setminus del_a \cup add_a$; G' contains all supersets of G.

Problem 7.2 (Partial Order Planning) 8 pt

Consider the planning task (P, A, I, G) where

- facts $P = \{a, b, c, d\}$
- actions $A = \{U, V, W\}$ where the preconditions (above the box) and effects (below the box) of the actions are given by

$$\begin{array}{ccc} a & b & a \\ \boxed{U} & \boxed{V} & \boxed{W} \\ b & \neg a, c & \neg a, d \end{array}$$

- initial state $I = \{a\}$
- goal $G = \{c, d\}$

Our goal is to build a partially ordered plan. Recall that each step is an action or the start step or the finish step.

1. Give the start step and finish step. 2 pt
2. Give all causal links between the steps. 2 pt
3. Give an example where a step clobbers a link. 2 pt
4. Give the temporal ordering that yields a partially ordered plan that solves the task. 2 pt

Solution:

1. $\underset{a}{\boxed{Start}} \quad \overset{c,d}{\boxed{Finish}}$

2. $Start \overset{a}{\rightarrow} U, Start \overset{a}{\rightarrow} W, U \overset{b}{\rightarrow} V, V \overset{c}{\rightarrow} Finish, W \overset{d}{\rightarrow} Finish$

3. Both steps V and W clobber both of the links $\overset{a}{\rightarrow}$.

4. $U \prec W$ and $W \prec V$