Name:

Birth Date:

Matriculation Number:

# Exam
# Artificial Intelligence 1

Feb 15, 2021

| | To be used for grading, do not write here | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| prob. | 1.1 | 2.1 | 2.2 | 2.3 | 3.1 | 3.2 | 4.1 | 4.2 | 5.1 | 5.2 | 6.1 | 6.2 | Sum | grade |
| total | 10 | 8 | 5 | 8 | 7 | 8 | 7 | 8 | 7 | 8 | 12 | 7 | 95 | |
| reached | | | | | | | | | | | | | | |

Exam Grade: Bonus Points: Final Grade:

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.
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.
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.

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 (Reading and Understanding Prolog)** 10 pt

Consider the query ?−animal(Y) over the following Prolog program:

```
foo([H|_], X) :− X=H.
foo([_|T], X) :− foo(T,X).

eats(mouse,fruit).
eats(dog,meat).
eats(cat,mouse).
eats(Y,X) :− Y=lion, alive(X).
eats(cat,bird).

animal(X) :− eats(X,Y), alive(X), food(Y).
animal(Y) :− Y=cat.

alive(Y) :− foo([mouse,cat,lion,bird], Y).
alive("dog").

food(X) :− alive(X).
food(fruit).
food(meat).
```

1. Which answers will the query return for Y? 5 pt
   - Only the **set** of answers matters — order or duplicates do not.
   - Write "none" if the query returns no answers.
2. Which answer is returned (a) first (b) second ("none" if no answer is returned). 2 pt
3. Add a rule to the program such that the above query does not return an answer. — 3 pt
   - Indicate exactly where you insert your rule.
   - You may **not** add a rule that is ill-formed (e.g., a syntax error, or calling a predicate that does not exist).
   - You may **not** use functions from standard library.

---

**Solution**:

1. mouse, cat, lion (Some of them are found multiple times, but we only asked for the set.)
2. (a) mouse
   (b) cat
3. Prolog's DFS always finds all solutions. So we need to
   - send DFS into an infinite loop by adding, e.g., animal(X):− animal(X), or
   - cut off the branches that have solutions by adding, e.g., animal(_):− !, false

   at the beginning.

---

## 2 Search

**Problem 2.1 (DFS and BFS Concretely)** 8 pt

Consider the **infinite** tree whose nodes are the natural numbers with root 0. For every node, the children and their order are as follows:

- children of 0: 1, 2, 3
- children of 1: 4, 5
- children of 2: 6, 7
- children of 3: 8
- children of 4: 9
- children of 5: 10
- children of 6: 11
- children of 9: 12
- children of $n$ for $n \geq 12$: n+1
- other nodes have no children

1. List the nodes in the order of expansion during
   (a) depth-first search 3 pt
   (b) breadth-first search 3 pt
2. Assuming the goal state is 7, why does it matter whether we use depth-first or breadth-first search? 2 pt

---

**Solution**:

1. (a) $0, 1, 4, 9, 12, 13, 14, \ldots$
   (b) $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, \ldots$
2. DFS will run forever without finding the goal state. BFS will find it.

---

**Problem 2.2 (Heuristics)** 5 pt

Consider heuristic search with heuristic function $h$.

1. Briefly explain what is the same and what is different between $A^*$ search and greedy search regarding the decision which node to expand next. 3 pt
2. Is the constant function $h(n) = 0$ an admissible heuristic for $A^*$ search? 2 pt

---

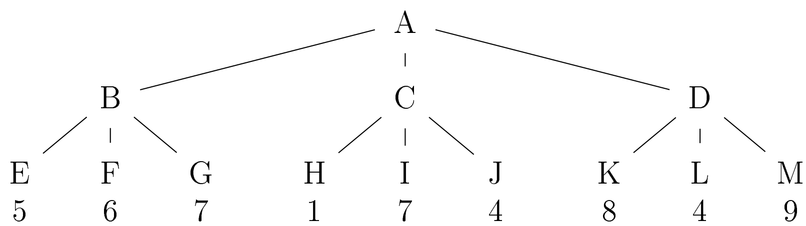
**Solution**:

1. Both choose the node that minimizes a certain function. As that function, $A^*$ uses the sum of path cost and heuristic whereas greedy only uses the heuristic.
2. Yes. (But it's a useless one.)

---

**Problem 2.3 (Adversarial Search)** 8 pt

Consider the following minimax game tree for the **maximizing player**'s turn. The values at the leaves are the static evaluation function values of those states.




1. Label each non-leaf node with its minimax value. 4 pt
2. Which move would be selected by the player? 1 pt
3. List the nodes that the alpha-beta algorithm would prune (i.e., not visit). Assume children of a node are visited left-to-right. 3 pt

---

**Solution**:

1. A=5, B=5, C=1, D=4
2. Move B
3. I, J, M

---

# 3 Constraint Satisfaction/Propagation

**Problem 3.1 (3 Rooks on a Small Board)** 7 pt

Consider the following problem: We want to place 3 rooks on a $4 \times 7$ chess-board such that no two rooks threaten each other. (Rooks move like queens except not diagonally.)
Model the problem as a constraint satisfaction problem $(V, D, C)$.
Explain your model briefly by saying how rook placements correspond to the assignments for the problem.

---

**Note:** 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**: $V = a_1, a_2, b_1, b_2, c_1, c_2$
$D_{a_1} = D_{b_1} = D_{c_1} = \{1, 2, 3, 4\}$
$D_{a_2} = D_{b_2} = D_{c_2} = \{1, 2, 3, 4, 5, 6, 7\}$
Constraints in $C$:

- $v_1 \neq w_1$ for all $(v, w) \in \{(a, b), (a, c), (b, c)\}$
- $v_2 \neq w_2$ for all $(v, w) \in \{(a, b), (a, c), (b, c)\}$

The assignments to $(a_1, a_2)$, $(b_1, b_2)$, and $(c_1, c_2)$ correspond to the coordinates of the squares where the rooks are placed.

---

**Problem 3.2 (CSP Formalization)** 8 pt

Consider the following binary CSP:

- $V = \{x, y, z\}$
- $D_x = \{0, 1, 2\}$, $D_y = \{1, 2\}$, $D_z = \{0, 1\}$
- Constraints: $x \neq y$, $y > z$

1. Give all pairs $(v, w)$ of variables such that $v$ is arc-consistent relative to $w$. 3 pt

2. Give all solutions that would remain if we added the constraint $x \neq z$. 3 pt

3. Assume we assign $y = 1$ and apply forward-checking. Give the resulting domains $D_x, D_y, D_z$. 2 pt

---

**Solution**:

- $(x, y)$, $(x, z)$, $(y, x)$, $(y, z)$, $(z, x)$, $(z, y)$
- Solutions $(x, y, z)$ are $(0, 2, 1)$, $(1, 2, 0)$, $(2, 1, 0)$
- $D_x = \{0, 2\}$, $D_y = \{1\}$, $D_z = \{0\}$

---

# 4 Logic

**Problem 4.1 (Satisfiability and Validity)** 7 pt

Prove or refute each of the following statements about first-order logic:

1. The formula $(\forall x.P(x) \vee Q(x)) \Rightarrow \big((\forall x.P(x)) \vee (\forall x.Q(x))\big)$ is satisfiable. 3 pt

2. If $A$ is not satisfiable, then $\neg A$ is valid. 4 pt

---

**Solution**:

1. True. A satisfying interpretation is any one that satisfies one of the following
   - Both $P$ and $Q$ false for some value,
   - $P$ is true for all values, or
   - $Q$ is true for all values.
2. True. $A$ not satisfiable implies all model-assignment pairs make $A$ false and thus make $\neg A$ true. So $\neg A$ is valid.

---

**Problem 4.2 (Proving in Natural Deduction)** 8 pt

Prove the following formula using natural deduction:

$$(P \Rightarrow Q) \Rightarrow \big(P \Rightarrow (P \wedge Q)\big)$$

---

**Solution**:

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{$Ax$}
\UnaryInfC{$P \Rightarrow Q, P \vdash P$}
\AxiomC{}
\RightLabel{$Ax$}
\UnaryInfC{$P \Rightarrow Q, P \vdash P \Rightarrow Q$}
\AxiomC{}
\RightLabel{$Ax$}
\UnaryInfC{$P \Rightarrow Q, P \vdash P$}
\RightLabel{$\Rightarrow E$}
\BinaryInfC{$(P \Rightarrow Q), P \vdash Q$}
\RightLabel{$\wedge I$}
\BinaryInfC{$P \Rightarrow Q, P \vdash P \wedge Q$}
\RightLabel{$\Rightarrow I$}
\UnaryInfC{$P \Rightarrow Q \vdash P \Rightarrow (P \wedge Q)$}
\RightLabel{$\Rightarrow I$}
\UnaryInfC{$\vdash (P \Rightarrow Q) \Rightarrow \big(P \Rightarrow (P \wedge Q)\big)$}
\end{prooftree}
$$

---

# 5 Knowledge Representation

**Problem 5.1 (Specifying Properties in ALC)** 7 pt

Consider the following ALC setting:

- concepts: `man`, `woman`,
- relations: `parentOf`, `spouseOf`

1. Give ALC expressions for
   (a) the concept of all persons in a homosexual marriage 2 pt
   (b) the statement that all men with children are married 2 pt
2. Translate to first-order logic the ALC statement $(\exists p.(m \sqcap w)) \sqsubseteq \forall s.\overline{m \sqcup w}$. 3 pt

---

**Note:** You may abbreviate every concept/relation by its first letter in your solution.

---

**Solution**:

1. (a) $(m \sqcap \exists s.m) \sqcup (w \sqcap \exists s.w)$
   (b) $(m \sqcap \exists p.\top) \sqsubseteq \exists s.\top$
2. $\forall x.\big(\exists y.p(x,y) \wedge m(y) \wedge w(y)\big) \Rightarrow \big(\forall y.s(x,y) \Rightarrow \neg(m(y) \vee w(y))\big)$

---

**Problem 5.2 (Induction on ALC)** 8 pt

Consider the following fragment of the ALC grammar for concepts $C$:

$$C ::= a \mid \top \mid C \sqcap C \mid \forall r.C$$

where $a$ and $r$ represent the names of atomic concepts and relations.
We define a substitution $s$ to be a mapping from atomic concepts $a$ to concepts $C$.

By induction on $C$, define a function $f$ such that $f(C,s)$ is the concept that arises from $C$ by substituting every $a$ with $s(a)$.

---

**Solution**:

$$\begin{aligned}
f(a,s) &= s(a) \\
f(\top,s) &= \top \\
f(C_1 \sqcap C_2, s) &= f(C_1,s) \sqcap f(C_2,s) \\
f(\forall r.C, s) &= \forall r.f(C,s)
\end{aligned}$$

---

# 6 Planning

**Problem 6.1 (Planning Deliveries in STRIPS)** 12 pt

Consider a truck that can carry 2 objects at a time and is supposed to deliver objects $Obj = \{V, W, X, Y\}$ from location $A$ to certain locations $Loc = \{A, B, C, D\}$ along some roads $Roads = \{\{A, B\}, \{B, C\}, \{B, D\}\}$. We use the following STRIPS task:

- facts: $\{\texttt{at}(l, o) \mid l \in Loc, o \in Obj\} \cup \{\texttt{truck}(l) \mid l \in Loc\}$
- actions $\texttt{move}(l, m, O)$ for $\{l, m\} \in Roads$, $O \subseteq Obj$, $|O| \leq 2$ given by
  - precondition: $\texttt{at}(l, o)$ for all $o \in O$, $\texttt{truck}(l)$
  - add list: $\texttt{at}(m, o)$ for all $o \in O$, $\texttt{truck}(m)$
  - delete list: same as precondition
- initial state: $\texttt{truck}(A)$, $\texttt{at}(A, o)$ for $o \in Obj$
- goal state: $\texttt{at}(C, V)$, $\texttt{at}(D, W)$, $\texttt{at}(B, X)$, $\texttt{at}(B, Y)$

1. Give an action that is applicable in the initial state and the resulting successor state. 4 pt
2. Give an optimal plan for the task above. 4 pt
3. Argue why the following heuristic $h$ is **not** admissible: for any state $s$, let $h(s)$ be the number of objects that are not at their goal location. 4 pt

---

**Solution**:

1. For any $O \subseteq Obj$ with $|O| \leq 2$, the action $\texttt{move}(A, B, O)$ results in the state containing $\texttt{truck}(B)$, $\texttt{at}(B, o)$ for $o \in O$, $\texttt{at}(A, o)$ for $o \notin O$.
2. $\texttt{move}(A, B, \{V, X\})$, $\texttt{move}(B, C, \{V\})$, $\texttt{move}(C, B, \{\})$, $\texttt{move}(B, A, \{\})$, $\texttt{move}(A, B, \{W, Y\})$, $\texttt{move}(B, D, \{W\})$
3. In the state where $\texttt{at}(C, V)$, $\texttt{at}(D, W)$, $\texttt{at}(A, X)$, $\texttt{at}(A, Y)$, $\texttt{truck}(A)$, the optimal plan $\texttt{move}(A, B, \{X, Y\})$ has length 1, but $h$ yields 2 in violation of the definition of admissible.

---

**Problem 6.2 (Partial Order Planning)** 7 pt

Consider partial-order planning.

1. Given a STRIPS task $\Pi := \langle P, A, I, G \rangle$, what are the components of a partially ordered plan? 4 pt
2. What are the conditions on a partially ordered plan to be complete and consistent? 2 pt
3. How can we turn such a plan into a solution of the original planning task? 1 pt

---

**Solution**:

1. A partially ordered plan consists of
   - a start node which has the facts in $I$ as a postcondition, 1 pt
   - a finish node which has the facts in $G$ as a precondition 1 pt

- causal links $S \xrightarrow{p} T$ where $p$ is a precondition fulfilled by $S$ 1 pt
- temporal ordering constraints $S \prec T$. 1 pt

2. A partially ordered plan is complete iff all preconditions are achieved by a causal link; and consistent iff the relation induced by causal links and ordering relations is a partial ordering. 2 pt

3. Any linearization of a complete partially ordered plan is a solution. 1 pt

---