Last Name: First Name:

Matriculation Number:

Exam
Artificial Intelligence 1

October, 2024

	To be used for grading, do not write here														
prob.	1.1	2.1	2.2	3.1	3.2	4.1	4.2	5.1	5.2	5.3	6.1	7.1	7.2	Sum	grade
total	10	12	8	5	3	9	7	6	6	8	6	6	5	91	
reached															

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.)

Problem 4.1 was changed to count for 10 points instead of 8. This changed the total points to 92 and the bonus points to 7.

1 Prolog

Problem 1.1 (Analyzing a Prolog Program)
Consider the following Prolog program:

```
1 rch(X,[],X).
2 rch(X,[A|As],Z) :- trns(X,A,Y), rch(Y,As,Z).
3
4 slt(As) :- i(X), g(Z), rch(X,As,Z).
```

1. Extend this program with clauses so that the query `slt([a,b])` succeeds. 3 pt

Solution: For example, `i(0). g(2). trns(0,a,1). trns(1,b,2).`
Technically, a solution like `slt(_)` is also correct. Except, technically, it is not because `i` and `g` are not defined. Such solutions received 2.5 points.

2. Which definition from the course does the program `slt(As)` implement? 3 pt

Solution: It checks if the list `As` of actions is a solution to the search problem whose initial/goal states are the ones satisfying `i` resp. `g` and which has transitions from state s to state s' under action a if trns(s, a, s') holds.

For the remaining questions, assume that the predicates `i`, `g`, and `trns` are defined by a finite set of facts. We now reverse the order of the subgoals in line 2, i.e., change line 2 to

```
rch(X,[A|As],Z) :- rch(Y,As,Z), trns(X,A,Y).
```

Explain (in about 2 sentences each) if/how this change affects the query `slt(As)` regarding

3. …correctness? 2 pt

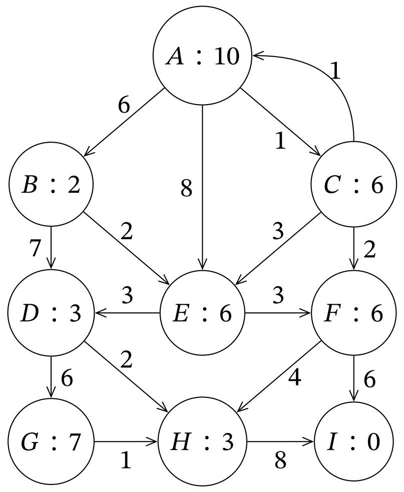
Solution: It has no effect. No clause has a side-effect, and the recursion in line 2 definitely terminates. Therefore, reordering subgoals does not change the behavior of the program.

4. …time efficiency? 2 pt

Solution: The program will now search backwards through all paths to `Z` instead of forwards through all paths from `X`. This has similar efficiency, and which method is faster depends on the structure of the search problem.

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.
Each node's children are ordered **alphabetically**.
Every edge is labeled with its actual cost.

1. Show or refute that the heuristic is admissible. 2 pt

Solution: It is not admissible: We have $h(A) = 10 \nleq 9 = h^*(A)$.
This assumes that I is the goal, which is implied but not clearly stated by the problem. Any answer arguing that the question depends on the choice of goal node is also accepted.

2. Give one example of a transition in that graph that makes the heuristic inconsistent. 2 pt

Solution: $A \to B$, $A \to C$, or $G \to H$ (because they decrease the heuristic by more than the cost)
Creating a new transition that would make the graph inconsistent is also accepted.

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**.

3. Depth-first search 1 pt

Solution: B, D, G, H

4. Breadth-first search 1 pt

Solution: B, C, E, D

5. uniform-cost search 2 pt

Solution: C, F, E, B

6. greedy search 2 pt

Solution: B, D, H, I

7. A^*-search 2 pt

Solution: C, B, F, I

Problem 2.2 (Search Problems)
Consider the family of search problems P_n given for $n = 1, 2, ...$ by $\langle S_n, A_n, T_n, I_n, G_n\rangle$ where

- $S_n = \{0,1\}^n$, i.e., states $x \in S_n$ are n-tuples $(x_1, \dots, x_n)$
- $A_n = \{l, r, i\}$
- T_n is given by
  - $T_n(l, x) = \{(x_2, \dots, x_n, 0)\}$
  - $T_n(r, x) = \{(0, x_1, \dots, x_{n-1})\}$
  - $T_n(i, x) = \{u\}$ where
    - if $x_n = 0$: $u = (x_1, \dots, x_{n-1}, 1)$
    - if $x_n = 1$: $u_n = 0$ and (if $n > 1$) $(u_1, \dots, u_{n-1}) \in T_{n-1}(i, (x_1, \dots, x_{n-1}))$
- $I = \{(0, \dots, 0)\}$
- $G = \{(1, \dots, 1)\}$

1. For $n = 5$, give the result of applying the action sequence i, l, i, i in the initial state. 2 pt

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

Note that the transition system is that of an n-bit register with action $\underline{l}$eft-shift, $\underline{r}$ight-shift, and $\underline{i}$ncrement.

2. What is the shortest solution to P_n? 2 pt

Solution: $i, l, \dots, i, l, i$ of length $2n - 1$.

There is also a suboptimal solution $i, \dots, i$ of length 2^n.

Now we try to solve P_n using depth-first search. We avoid cycles by only applying an action if it leads to a previously unexpanded state.
Discuss this strategy for the particular case of P_n (in about 1-2 sentences each) regarding

3. …completeness. 2 pt

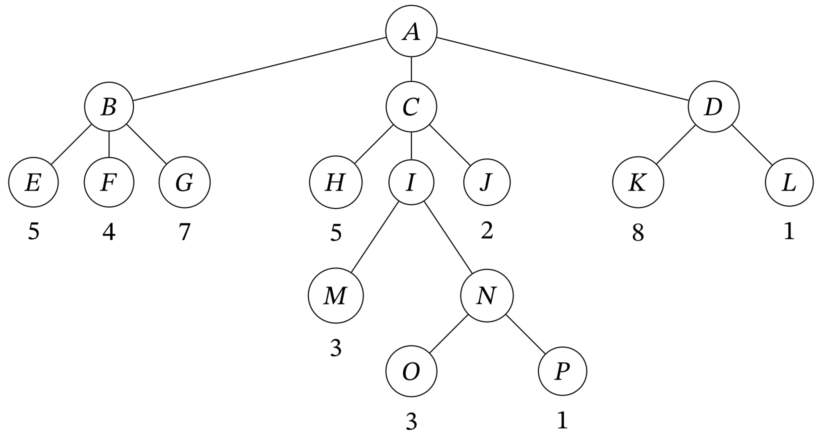
Solution: Complete. Because the state space is finite and the goal reachable, it always finds a solution.

4. optimality. 2 pt

Solution: Optimality behavior depends on the unspecified order in which we try actions. For example, trying i first finds the suboptimal solution $i, \ldots, i$. DFS will not find the optimal solution using any order because it would require alternating l and i even though both actions are always applicable.

3 Adversarial Search

Problem 3.1 (Minimax)
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; some of those values are currently missing.



1. Label the node A with its minimax value. 2 pt

Solution: 4

2. Which move would be chosen by the player? 1 pt

Solution: B

3. Which nodes does $\alpha\beta$-pruning prune? We expand child nodes in alphabetical order. 2 pt

Solution: P, J

Problem 3.2 (Minimax Applicability)
Consider the following game:

- Initially, 2 players have 10 tokens each, and there is an empty bag of tokens in the center.

- The players take turns either putting an odd number of tokens into the bag or taking a non-zero even number of tokens from the bag.
- A player loses if they have no more tokens.

1. Explain why minimax can/cannot be used to find a perfect strategy for this game. 3 pt

   Solution: It cannot be used. The game is not guaranteed to be finite, e.g., the move sequence put 1, put 1, take 2 could repeat forever.

4 Constraint Satisfaction

Problem 4.1 (Solving and Propagation)
Consider the following binary CSP:

- $V = \{a, b, c, d, e\}$
- $D_a = D_b = D_c = \{0, 1, 2, 3\}, D_d = D_e = \{0, 1, 2, 3, 4, 5\}$
- Constraints:
  - $a > 1$ or $d > 2$
  - $b \cdot e > 8$
  - $d = 2c$
  - $a > b$
  - $b > c$

1. Give an **inconsistent** total assignment to the variables. 1 pt

   Solution: Any assignment that is not a solution, e.g., $a = b = c = d = e = 0$.

2. Give **the two** solutions. 3 pt

   Solution: $a = 3$, $b = 2$, $(c, d) = (0, 0)$ or $(c, d) = (1, 2)$, $e = 5$

3. Check the boxes for (v, w) where v is arc-consistent relative to w. 2 pt

   ☐ (a, d) ☐ (d, a) ☐ (c, d) ☐ (d, c)

   Solution: $(a, d), (d, a)$

4. Now assume you have assigned $b = 2$. Give the domains after forward-checking. 2 pt

   Solution: $D_a = \{3\}, D_c = \{0, 1\}, D_d = \{0, 1, 2, 3, 4, 5\}, D_e = \{5\}$

5. What additional step can be taken if forward-checking results in a variable domain of size 1? 1 pt

Solution: The variable can be assigned to its unique remaining value, and forward-checking can be called again.

Problem 4.2 (Relating CSP and SAT)
We want to reduce SAT to CSP.

1. Explain (in 2 sentences) why it is easier and sufficient to reduce SAT to higher-order CSPs (as opposed to reducing SAT to binary CSPs). 2 pt

Solution: Easier: Higher-order CSPs are a larger class. So the reduction is easier. Sufficient: We already know that higher-order CSPs can be reduced to binary CSPs.

2. Show that SAT can be reduced to CSP by defining 5 pt
   - for every SAT-instance P a CSP-instance $Q = (V, D, C)$ and
   - a bijection between Q-solutions and satisfying P-instances

Solution: Assume P given by a propositional signature $\Sigma = \{X_1, \ldots, X_n\}$ and a Σ-formula F.
We define $Q = (V, D, C)$ by
- $V = \Sigma$
- $D_X = \{0, 1\}$ for all $X \in V$
- C contains only $F = 1$ where F is seen as a Boolean function.

Both satisfying assignments for P and solutions of Q are maps $\Sigma \to \{0, 1\}$, and the bijection is the identity.

5 Logic

Problem 5.1 (Propositional Logic)
Consider the formula $A = (p \wedge q \wedge \neg r) \vee (\neg p \Rightarrow r) \vee (p \wedge q \wedge r)$ using propositional variables p, q, r.

1. Give all falsifying assignments for A. 2 pt

Solution: $\langle \varphi(p), \varphi(q), \varphi(r) \rangle \in \{\langle F, F, F \rangle, \langle F, T, F \rangle\}$

2. Give a formula in CNF that is equivalent to A. 2 pt

Solution: $p \vee r$ (This can be read off off the list of falsifying assignments. Reading off a faulty list was also accepted.)

3. Give a formula B such that $A \vee (B \Rightarrow A)$ is valid. 2 pt

Solution: $B = false$ and $B = A$ are the easiest options.
But any formula that is false for the two falsifying assignments works.

Problem 5.2 (Modeling in First-Order Logic)
Consider the following situation:

- Some individuals are persons, some are animals.
- Individuals may own other individuals.

1. Model this situation in first-order logic by giving a signature, i.e., a list of function/predicate symbols with arity. 2 pt

Solution:

- unary predicate symbols: p (for person), a (for animal)
- binary predicate symbols: o (for owns)

2. State formulas over your signature that capture the following properties: 2 pt
   1. There is a person that owns an animal.
   2. Persons cannot be owned.

Solution:

1. $\exists x.\exists y.p(x) \wedge a(y) \wedge o(x, y)$
2. $\forall x.p(x) \Rightarrow \neg\exists i.o(i, x)$

3. Give a model of your signature in which the universe is as small as possible and the above two formulas are true. 2 pt

Solution: $D = \{Alice, Bubble\}$, $I(p) = \{Alice\}$, $I(a) = \{Bubbles\}$, $I(o) = \{(Alice, Bubbles)\}$.

Problem 5.3 (Resolution)
Consider the signature of first-order logic with unary predicate symbol P, binary predicate symbol R, and nullary function symbols a, b.

1. Prove the following formula using the first-order resolution calculus. 8 pt

$$\exists X.\forall Y.\exists Z.\exists W.(\neg P(Z) \wedge \neg R(b, a)) \vee \neg R(a, b) \vee R(W, a) \vee (P(Y) \wedge R(X, b))$$

Solution: We negate:

$$\forall X.\exists Y.\forall Z.\forall W.(P(Z) \vee R(b, a)) \wedge R(a, b) \wedge \neg R(W, a) \wedge (\neg P(Y) \vee \neg R(X, b))$$

We skolemize:

$$(P(Z) \vee R(b, a)) \wedge R(a, b) \wedge \neg R(W, a) \wedge (\neg P(f_Y(X)) \vee \neg R(X, b))$$

This yields the clauses $\{P(Z)^{\mathsf{T}}, R(b,a)^{\mathsf{T}}\}, \{R(a,b)^{\mathsf{T}}\}, \{R(W,a)^{\mathsf{F}}\}, \{P(f_Y(X))^{\mathsf{F}}, R(X,b)^{\mathsf{F}}\}$. We resolve:

$$\{P(Z)^{\mathsf{T}}, R(b,a)^{\mathsf{T}}\} + \{R(W,a)^{\mathsf{F}}\}[b/W] \Longrightarrow \{P(Z)^{\mathsf{T}}\}$$
$$\{R(a,b)^{\mathsf{T}}\} + \{P(f_Y(X))^{\mathsf{F}}, R(X,b)^{\mathsf{F}}\}[a/X] \Longrightarrow \{P(f_Y(a))^{\mathsf{F}}\}$$
$$\{P(Z)^{\mathsf{T}}\}[f_Y(a)/Z] + \{P(f_Y(a))^{\mathsf{F}}\} \Longrightarrow \emptyset$$

6 Knowledge Representation

Problem 6.1 (ALC)
Consider the following ALC signature

- concept symbols: p (for person), a (for animal)
- role symbols: o (for owns)

1. Give a concept that represents the set of people that own animals. 1 pt

Solution: $p \sqcap \exists o.a$

2. Give an axiom that states that people cannot be owned. 2 pt

Solution: $\exists o.p \sqsubseteq \bot$

3. Give an ABox that represents a person Alice owning an animal Bubbles. 1 pt

Solution: $Alice : p, Bubbles : a, Alice\ o\ Bubbles$

4. Calculate the translation to first-order logic of the concept $\forall o.\exists o.a$. 2 pt

Solution: $\forall x.o(u,x) \Rightarrow \exists y.o(x,y) \wedge a(y)$ with free variable u

7 Planning

Problem 7.1 (STRIPS and Relaxation)
Consider the STRIPS task given by

- facts: $\{a,b,c,d\}$
- actions: $\{p,q,r\}$ with $p = \langle\{a\},\{b\},\{a\}\rangle$, $q = \langle\{c\},\{d\},\{c\}\rangle$, and $r = \langle\{b\},\{c\},\{b\}\rangle$, where each action is given as a triple of preconditions, add list, and delete list
- initial state: $\{a\}$

- goal state: $\{d\}$

1. Give the optimal plan for this problem. 2 pt

Solution: p, r, q

2. Give the optimal delete-relaxed plan for this problem. 2 pt

Solution: p, r, q

3. Give the optimal only-adds-relaxed plan for this problem. 2 pt

Solution: q

Problem 7.2 (Kinds of Planning)

1. Explain (in about 3-5 sentences) the differences between satisficing, optimal, and relaxed planning. 5 pt

Solution: Optimal planning tries to find the shortest plan.

Satisficing planning tries to find any plan, not necessarily the shortest. It is easier than optimal planning.
Relaxed planning finds a plan for a simplified problem. That does not solve the original problem, but it is easier and can be used as a heuristic.