WeST People and Knowledge Networks

UNIVERSITÄT KOBLENZ · LANDAU

Artificial Intelligence 1 SS 2022

Main Exam

Last name, first name	
Student ID	

To be completed by the corrector:

Assignment	1	2	3	4	5	6	7	Sum
Corrector								
Points	35	12	32	23	20	20	20	162
Achieved								

Grade

- [ ] Very good
- [ ] Good
- [ ] Fair
- [ ] Sufficient
- [ ] Failed

As a number:

Check whether your exam is complete (it should contain 7 assignments). Read each assignment *completely* before starting to answer it. **Do not use a pencil!** If you need additional pages, please write your name and student id on them.

Good luck!

Dr. Claudia Schon

1 Classical logics (35 Points)

a) (20 points)

Formalize the following sentences in first-order logic. Use the following predicate names with arity one

abraxan, whiskey, ability, fly, witch, wizard, man, human, horse, magic, muggle, squib,

the constants

harry, ron, molly

and the following predicates with arity two

hasChild, hasAbility, drinks.

(i) Wizards are men. (2 points)

(ii) Harry is a wizard. (2 points)

(iii) Ron is Molly's child. (2 points)

(iv) Witches have a magical ability. (3 points)

(v) A squib is a human without magical abilities, but who has (at least) one parent who is a witch or wizard. (3 points)

(vi) A muggle is a human who has no magical abilities and both his or her parents are neither a witch nor a wizard. (3 points)

(vii) An Abraxan is a flying horse that only drinks whiskey. (3 points)

(viii) Horses are not humans. (2 points)

Solution

a) $\forall X(wizard(X) \Rightarrow man(X))$ (2 points)

b) $wizard(harry)$ (2 points)

c) $hasChild(molly, ron)$ (2 points)

d) $\forall X(witch(X) \Rightarrow (\exists Y(ability(Y) \wedge magic(Y) \wedge hasAbility(X,Y))))$ (3 points)

e) $\forall X(squib(X) \Rightarrow (human(X) \wedge \neg\exists A(ability(A) \wedge magic(A) \wedge hasAbility(X,A)) \wedge \exists Y(hasChild(Y,X) \wedge (witch(Y) \vee wizard(Y)))))$ (3 points)

f) $\forall X(muggel(X) \Rightarrow (\neg\exists A(ability(A) \wedge magic(A) \wedge hasAbility(X,A)) \wedge \forall P(hasChild(P,X) \Rightarrow (\neg witch(P) \wedge \neg wizard(P)))))$ (3 points)

g) $\forall X(abraxan(X) \Rightarrow (horse(X) \wedge \exists F(ability(F) \wedge fly(F) \wedge hasAbility(X,F)) \wedge \forall D(drinks(X,D) \Rightarrow whiskey(D))))$ (3 points)

h) $\forall X(horse(X) \Rightarrow \neg human(X))$ (2 points)

b) (10 points)

Let α, β and γ be propositional formulae. For each of the following statements, say whether it is true or false. Please note: For each correct cross you get 2 points. For each incorrect cross 2 points are subtracted. In total, there is never less than 0 points for this subtask.

true	false	
☐	☐	$\{e \Rightarrow f, e \lor \neg a, a \lor b\} \vdash a \Rightarrow f$.
☐	☐	$\{a \Rightarrow b, \neg a\} \vdash \neg b$.
☐	☐	$\{a \lor b, \neg b \lor c, \neg c \lor a\} \vdash a$.
☐	☐	$\{e \lor \neg c, \neg b \lor \neg e, a, a \Rightarrow (a \land b)\}$ is satisfiable.
☐	☐	If $\alpha \vdash \beta$ and $\beta \vdash \gamma$ than $\alpha \vdash \gamma$.

Solution

2 points each
true - false - true - true - true

c) (5 points)

Explain why

$$\forall X : (hasChild(harry, X) \Rightarrow wizard(X)) \nvdash \exists X : (hasChild(harry, X) \land wizard(X)).$$

Do this by finding an interpretation I that is a model of $\forall X : (hasChild(harry, X) \Rightarrow wizard(X))$ but not of $\exists X : (hasChild(harry, X) \land wizard(X))$. Remember to state all necessary parts of the tuple $I = (U_I, f_I^U, P_I, F_I)$.

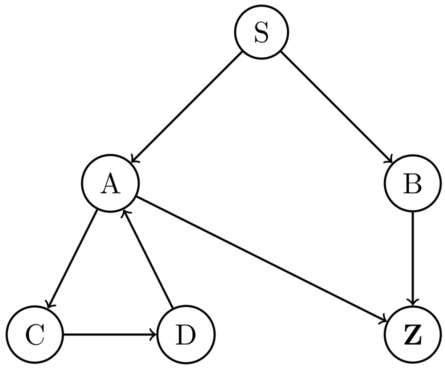
Solution

E.g. the model $I = (U_I, f_I^U, P_I, F_I)$ defined by:

- $U_I = \{a\}$
- $f_I^U(harry) = a$
- $P_I = \{hasChild^I, wizard^I\}$ with $hasChild^I = \emptyset$, $wizard^I = \emptyset$
- $F_I = \emptyset$

2 Search Problems (12 Points)

We consider the following search space with start node S and goal node Z.



For each search strategy listed below, specify the order in which the nodes are taken from the frontier. Specify as many nodes until the goal is reached or you have listed 8 expanded nodes (S is not counted, so you should add 8 nodes).

Depth First Search

Order: S, ______________________

Solution

Order: S, A, C, D, A, C, D, A, C (4 points)

Breadth First Search

Order: S, ______________________

Solution

Order: S, A, B, C, Z (4 points)

Iterative Depth First Search (with inkrement 1)

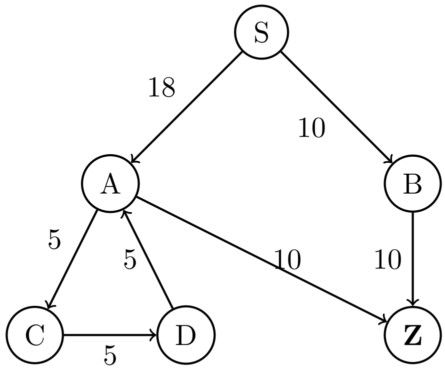
Order: S, ______________________

Solution

Reihenfolge: S, S, A, B, S, A, C, Z (4 points)

3 $A^\star$ Search (32 Points)

We consider the following search space with start node S and target node Z.



The following table defines the heuristic functions $h_0(x)$ to $h_3(x)$.

	S	A	B	C	D	Z
$h_0(x)$	19	0	9	19	14	0
$h_1(x)$	20	10	10	20	15	0
$h_2(x)$	21	11	11	21	16	0
$h_3(x)$	0	0	0	0	0	0

a) (8 points)
For each of these functions, state whether it is an admissible heuristic for the $A^\star$ algorithm. In each case, give a brief justification for your statement (two sentences maximum).

Solution

Admissible heuristics are h_0, h_1, and h_3, since they *underestimate* the cost. Not admissible is heuristic h_2, since it overestimates the cost (e.g., $h_2(S) = 21$, although the actual cost from S to the destination is only 20).
2 points for each heuristic

b) (4 points)
Which of the above heuristics is best suited to find a path to the goal using the $A^\star$ algorithm? Justify briefly (maximum 5 sentences)!

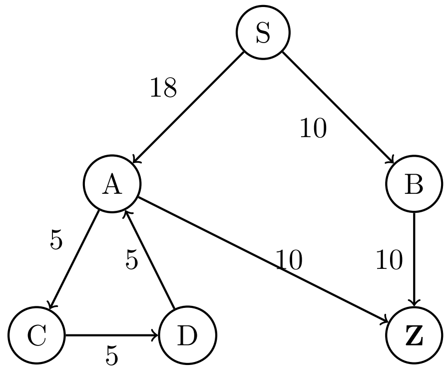
Solution

Of the admissible heuristics h_0, h_1 and h_3, h_0 dominates the heuristic h_3 and h_1 dominates the heuristic h_0 (and thus h_3). This means that the values of h_1 for all nodes are larger than those of h_0 and h_3. Thus, h_1 is closer to the actual cost than h_0 and h_3. Since a better running time of the $A^\star$ algorithm can be expected for h_1, h_1 should be chosen.

c) (20 points)

Use the $A^\star$ algorithm together with heuristic $h_0(x)$ to find the shortest path between S and Z. Use one row of the following table for each step, starting with step 2. Always write down the node (together with the path to the node) and its value according to the state evaluation function f. If there are two nodes with the same value for f in the frontier, select the one which was added last to the frontier. Circle the node that is being taken out of the frontier in each step. Write down the path that the algorithm found. Also state whether the path is optimal. You can assume that each node is only expanded once in each path, meaning that we assume a duplicate detection. Of course, it is still possible for A* to find different paths to a node.

Here again the search space



and the definition of h_0:

	S	A	B	C	D	Z
$h_0(x)$	19	0	9	19	14	0

Step	Frontier			
1	S = 0 + 19 = 19			
2	SA =	SB =		
3				
4				
5				
6				

Order in which the nodes are taken from the frontier: ____________________

Found path: ____________________

Costs of the path: ____________________

Is the found path optimal? Yes ☐ No ☐

Solution

Frontier development (sorted in ascending order by f values):
S = 0+19 = 19
SA =18+0=18, SB=10+9=19
SB=10+9=19,SAZ=28+10=28,SAC=23+19=42
SBZ=20+0=20,SAZ=28+10=28,SAC=23+19=42
The nodes which are taken out of the frontier are printed in bold face.
2 points for each correct cell in the table.

The order in which the nodes are taken from the frontier is: S, A, B, Z (1 point)
Found path: SBZ (1 point)
Costs of the path: 20 (1 point)
Optimal: yes (1 point)

4 STRIPS (23 Points)

Two freight trains run between the German towns *Neuwied* and *Andernach*. The trains are able to move from one town to another using the `drive` action. Both towns have a depot containing goods. Initially, goods g_1 and g_2 are both situated in Andernach. Using the action `load`, goods can be taken out of a depot and loaded on a train. Using the action `unload`, goods can be removed from a train and put into a depot. Note that only empty trains can load goods and each train can only load one good at a time. Further the actions `load` and `unload` change which goods are in the depot of a town.

The goal is to have both goods situated in Neuwied.

The following actions can be used:

- `drive(T,S1,S2)`: `T` drives from `S1` to `S2`.
- `load(T,G)`: `T` loads the `G`.
- `unload(T,G)`: `T` unloads the `G`.

To model this we use the following constants to denote the objects in our world: t_1 and t_2 for the two trains, *neuwied* and *andernach* for the two towns, and g_1 and g_2 for the two goods.

Furthermore, the following predicates are given:

- `in(X,Y)`: `X` is situated in town `Y`. This can be used to indicate that trains or goods are situated in a town.
- `loaded(T,G)`: `T` is loaded with `G`.
- `empty(X)`: `X` is empty.
- `train(X)`: `X` is a train.
- `town(X)`: `X` is a town.
- `good(X)`: `X` is a good.

a) Describe the initial state and the goal using the predicates and constants above. For the two trains you are free to choose in which towns they are initially situated.

Initial state: ____

____

____

Goal: ____

b) Describe the actions $drive(Z, X, Y)$, $load(T, G)$, and $unload(T, G)$ using STRIPS notation by specifying the action's (pre-)conditions C, delete-list D, and add list A:

drive(Z,X,Y): C: ______

D: ______

A: ______

load(T,G): C: ______

D: ______

A: ______

unload(T,G): C: ______

D: ______

A: ______

Solution

a) Initial state: $\{town(neuwied), town(andernach), train(t_1), train(t_2), in(t_1, neuwied), in(t_2, andernach), good(g_1), good(g_2), in(g_1, andernach), in(g_2, andernach)\}$. (3 points)
If it is $empty(t_1)$ or $empty(t_2)$ is stated, this is not a mistake.

Goal: $\{in(g_1, neuwied), in(g_2, neuwied)\}$. (2 points)

b) drive(Z,X,Y):
C: train(Z), town(X), town(Y), in(Z,X)
D: in(Z,X)
A: in(Z,Y)

load(T,G):
C: train(T), empty(T), good(G), in(G,X), in(T,X)
D: in(G,X), empty(T)
A: loaded(T,G)

unload(T,G):
C: train(T),loaded(T,G),good(G), in(T,X)
D: loaded(T,G)
A: empty(T), in(G,X)

2 points for each correct list of preconditions, add-list and delete-list. So at most 6 points for each action.

5 Default theory (20 Points)

Let $T = (W, \Delta)$ be a default theory with $W = \emptyset$ and $\Delta = \{\delta_1, \delta_2, \delta_3\}$ with

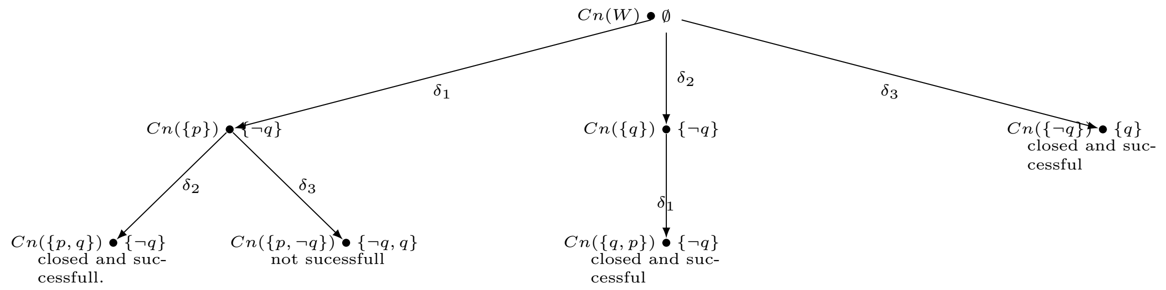
$$\delta_1 = \frac{\top \ : \ q}{p}, \quad \delta_2 = \frac{\top \ : \ q}{q}, \quad \delta_3 = \frac{\top \ : \neg q}{\neg q},$$

Calculate all extensions of T using a process tree.

$\mathsf{Cn}(W) \quad \bullet \quad \emptyset$

Extensions: ____

Solution



Extensions are: $Cn(\{p, q\}), Cn(\{\neg q\})$

- 2 points per node in the tree,
- 1 point for the decision if a leave is successful and closed or unsuccessful, and
- 2 points for each extension.

6 Answer set programming (20 Points)

Consider the following extended logic program P:

$$
\begin{array}{ll}
P: & a \leftarrow s. \\
 & \neg t \leftarrow not\ d, not\ e. \\
 & d \leftarrow r, not\ \neg t. \\
 & s \leftarrow not\ e. \\
 & f \leftarrow a, d. \\
 & g \leftarrow a, s. \\
 & r. \\
 & y.
\end{array}
$$

Find all answer sets of P. For each answer set, provide the reduct and justify why it is an answer set. *Hint*: There are exactly two answer sets.

Solution

10 points per answer set (4 per state, 4 per reduct, 1 for minimal model which is equal to the state),

There are two models:

- $S_1 = \{s, a, r, g, y, d, f\}$ with reduct

$$\begin{aligned} P^{S_1} = \{ & a \leftarrow s. \\ & d \leftarrow r. \\ & s. \\ & f \leftarrow a, d. \\ & g \leftarrow a, s. \\ & r. \\ & y.\} \end{aligned}$$

  und minimalem Modell $minModell(P^{S_1}) = \{s, a, r, g, y, d, f\}$.

- $S_2 = \{s, a, r, g, y, \neg t\}$ with reduct

$$\begin{aligned} P^{S_1} = \{ & a \leftarrow s. \\ & \neg t. \\ & s. \\ & f \leftarrow a, d. \\ & g \leftarrow a, s. \\ & r. \\ & y.\} \end{aligned}$$

  und minimalen Modell $minModell(P^{S_2}) = \{s, a, r, g, y, \neg t\}$.

7 Prolog (20 Points)

Consider the following prolog program:

```
p(e).
q(d).
r(d).
s(d).
q(a).
r(a).
s(a).
w(b).

p(X) :- q(X),
        r(X),
        !,
        s(X).

p(X) :- w(X).
```

Fill in the following table. Give **all** answers that prolog provides (also by typing ;). Specify the outputs in the order in which they are given by prolog.

Query	Output
p(X)	
p(a)	
p(X),p(Y)	

Solution

4 points per correctly filled table cell.

Query	Output
p(X)	X=e; X=d
p(a)	true
p(X),p(Y)	X=Y, Y=e; X=e, Y=d; X=d, Y=e; X=Y, Y=d

Let us now imagine the above program **without** the cut (`!`). The resulting prolog program looks as follows:

```
p(e).
q(d).
r(d).
s(d).
q(a).
r(a).
s(a).
w(b).

p(X) :- q(X),
        r(X),
        s(X).

p(X) :- w(X).
```

What output do we then get in response to the following queries? Again, give all the outputs in exactly the order that Prolog returns (also by typing `;`).

Query	**Output**
`p(X)`	
`p(a)`	

Solution

4 points per correctly filled table cell.

Query	**Output**
`p(X)`	X=e; X=d; X=a; X=b
`p(a)`	true