MASTER IN ARTIFICIAL INTELLIGENCE (UDC - USC - UVigo)
REASONING AND PLANNING exam. January 8th, 2024

Surname: _____

First Name: _____

INSTRUCTIONS This exam covers units 1-6 and is weighted with a maximum of **42 points (pt) from a total of 100 pt** in the whole course (Unit 7 is not covered in the exam and weights 8 pt). For the test, use the original statement sheet and avoid corrections or unclear marking (ask for a new blank sheet if needed). **Completion time = 2 hours**.

— EXAM —

Exercise 1 (20pt). Each question has at least one correct answer and its total score depends on whether you check: some incorrect answer = **-3pt**; all the correct answers = **5pt**; only correct answers, but not all = **3pt**; leaving blank = **0pt**. A total negative score in Exercise 1 counts as 0pt in the rest of the exam.

1.1) Mark those formulas below that are stronger than $p \rightarrow \neg q$ in classical propositional logic:

- [x] $p \wedge \neg q$
- [ ] p
- [x] $\neg q$
- [x] $p \wedge \neg p$

1.2) The logic program P with rules `p :- not q, r.` `r :- p.` is stratified. Mark the rules below that, if they were (individually) added to P, they would make the result a non-stratified program.

- [ ] `p :- r.`
- [x] `q :- not p.`
- [x] `q :- r.`
- [ ] `r :- q.`

1.3) Given the following logic program `p :- q.` `p :- not p, not q.`

- [x] the reduct with respect to $\{q\}$ is the program `p :- q.`
- [ ] the reduct with respect to $\{q\}$ is the program `p :- q.` `p :- not p.`
- [x] the reduct with respect to $\emptyset$ is the program `p :- q.` `p.`
- [ ] the reduct with respect to $\{p\}$ is the program `p :- q.` `p.`
- [ ] the reduct with respect to $\{p, q\}$ is the program

1.4) The rule `p :- not p, not q.` used above is actually equivalent to the formula $\neg\neg p \vee \neg\neg q$ in the logic of Here-and-There (HT), but the latter is not equivalent to $p \vee q$ in that logic. Mark those HT interpretations that are HT models of $\neg\neg p \vee \neg\neg q$ but not of $p \vee q$.

- [x] $H = \emptyset$, $T = \{p\}$
- [ ] $H = \{q\}$, $T = \{q\}$
- [x] $H = \emptyset$, $T = \{p, q\}$
- [ ] $H = \{p, q\}$, $T = \{p\}$

Explanations for the test

1.1) The formula $p \to \neg q$, call it α, has only one countermodel $\{p, q\}$ (when the antecedent is true and the consquent false).

- This formula, $p \wedge \neg q$, only has one model $\{p\}$ which makes α true.
- The formula p has two models, but one of them $\{p, q\}$ is a countermodel of α.
- The formula $\neg q$ makes α true (it makes true its consequent)
- The formula $p \wedge \neg p$ has no models so it is the strongest possible formula. All models of $p \wedge \neg p$ are models of α, because $p \wedge \neg p$ has no models.

1.2) The program graph has three dependences: p depends negatively on q; p depends positively on r; and r depends positively on p.

- Adding `p:-r` does not change the program dependences (p still depends positively on r).
- Adding `q:-r` creates a negative loop: p depends negatively on q, which depends on r which in its turn depends on p.
- Adding `q:-not p` creates a negative loop: p depends negatively on q, and q depends negatively on p.
- Adding `r:-q` does not create any new loop.

1.3) Answer number 2 is incorrect: the program reduct never contains a negation (all `not`'s are removed). On the other hand, any positive rule is preserved untouched in the reduct: this applies to the first rule `p :- q`, that must always be present in all reducts, so answer 5 is also incorrect. On the other hand, as soon as we have p or q in the assumption, the second rule must be removed, so answer 4 is incorrect. The other two answers are correct.

1.4) Answer 4 is obviously incorrect, since an HT interpretation must satisfy $H \subseteq T$ by construction. The models of the formula $\neg\neg p \vee \neg\neg p$ are those where in the assumed atoms T we have either p or q or both, but it does not impose any restriction on the proved atoms H. Thus, answers 1, 2, 3 are all models of $\neg\neg p \vee \neg\neg p$. However, to be a model of $p \vee q$ we must additionally have either p or q (or both) proved in H. So answers 1 and 3 are not models of $p \vee q$.

Exercise 2 (10pt). Write an ASP program that generates all ways to place 5 rooks in a chessboard so that they do not attack each other. Use predicate `rook(X,Y)` meaning there is a rook at row `X` and column `Y`. (NOTE: in chess, rooks attack other pieces in the same row or in the same column).

```
#const n=8.
cell(1..n,1..n).
5 {rook(X,Y): cell(X,Y)} 5.
:- rook(X,Y), rook(X,Y'), Y!=Y'.
:- rook(X,Y), rook(X',Y), X!=X'.
#show rook/2.
```

Exercise 3 (8pt). The following `telingo` program tries to move a robot in a grid from an initial position at `(0,0)` to a goal position at `(3,4)`. Complete the program to fulfil the two missing requirements: (1) the robot must reach the goal position at the end; (2) the robot cannot step into a wall.

```
#program initial.
grid(0..3,0..4).
wall(0,2).  wall(2,2).   wall(3,2).   robot(0,0).   goal(3,4).

#program dynamic.
1 { robot(X+1,Y); robot(X-1,Y); robot(X,Y+1); robot(X,Y-1) } 1 :- 'robot(X,Y).
:- robot(X,Y), not _grid(X,Y).  % Do not step out of the grid
:- robot(X,Y), _wall(X,Y).      % Do not step into a wall

#program final.
:- robot(X,Y), not _goal(X,Y).  % Reach the goal at last state
```

Exercise 4 (4pt). Write a formula in Description Logic (DL) that describes the set of individuals with children (use relation *has_child*) being all of them students (use concept name *Student*).

$$\exists \mathit{has\_child} \sqcap \forall \mathit{has\_child}.\mathit{Student}$$