MODULE 11 - FAI - EXERCISES

1

1) We have to encode a knowledge in which we have concepts and roles (Father, Mother...) and relations between them.
We need QUALIFIED NUMBER RESTRICTIONS (because in the description of the knowledge we have "at least 3", "at most 2"...), therefore I think that ALCQ could be a good KRL in this situation.

2) In this point of the exercise we need to express which are the BASE (PRIMITIVE) CONCEPTS (those for which we don't express the definition in the TBOX)
we have also to express BASE ROLES.

- So, to correctly encode our knowledge, we should assume and encode concept such as Male, Female, Person.
(these concepts are primitive because we cannot describe the concept "female" for example and it's not in the scope of the exercise.
These are important primitive concepts because allow us to DEFINE AND DIFFERENTIATE concepts in our knowledge, such as Mother, Father etc...)

- BASE ROLES:
  - hasChild
  - hasSister
  - hasBrother

- It could also be useful to define INVERSE RELATION between:
  - sisterOf and hasSister
  - brotherOf and hasBrother
  - childOf and hasChild.

- Therefore I suggest to use ALCQI as KRL.

3. Write the KB in the chosen KRL.

- We do not have the A-Box in our case because in the description of the knowledge there are not instances of the concepts.
- We define the TBOX only.

TBOX: {

$Father \equiv Male \sqcap \exists hasChild.Person,$

$Mother \equiv Female \sqcap \exists hasChild.Person,$

$Parent \equiv Father \sqcup Mother,$

$GrandMother \equiv Female \sqcap \exists hasChild.Parent,$

$GrandFather \equiv Male \sqcap \exists hasChild.Parent,$

Ruolo attivo: ha una sorella

$Aunt \equiv Female \sqcap (\exists hasSister.Mother \sqcup \exists hasBrother.Father)$

la nonna è una femmina che ha almeno un figlio il cui ruolo è "genitore".

Se non avessi definito Parent avrei scritto:

$GrandMother \equiv Female \sqcap (\exists hasChild.Mother \sqcup \exists hasChild.Father)$

$Uncle \equiv Male \sqcap (\exists hasSister.Mother \sqcup \exists hasBrother.Father)$

$ChildOf \equiv inverse(hasChild),$

Ruolo passivo: è figlia di

$Niece \equiv Female \sqcap (\exists ChildOf.(Parent \sqcap (hasSister.Parent \sqcup hasBrother.Parent)),$

la zia è una femmina che ha almeno una sorella che è mamma o un fratello che è padre

$Nephew \equiv Male \sqcap (\exists ChildOf.(Parent \sqcap (hasSister.Parent \sqcup hasBrother.Parent)),$

$Mother\_of\_AL\_3 \equiv Female \sqcap \geq 3\, hasChild.Male;$

(sons) maschi

$Father\_of\_AM\_3\_D \equiv Male \sqcap \leq 3\, hasChild.Female$

daughters (femmine)

}

4) Queries to the KB.

$Mother \sqsubseteq Parent$?

$GrandFather \sqsubseteq Male$?

2

1) For this knowledge we can use FOL (as KRL), since we do not need the additional features of DL or SC (situation calculus)

(• we don't need to express complex roles or relations between concepts for which we need definition - we do not have a series of Names such as Father, Mother... which need to be defined ⇒ NO DL

• we do not have a situation that changes in time (non STATIC DOMAIN) ⇒ NO SC

• We have a series of sentences universally and existentially quantified ⇒ FOL)

2) The exercise asks us to assume a set of individuals (invented by us)

• Let's assume a set of individuals for all the possible humans and animals:

· Human (Carlo) · Dog (Fuffy) · ¬WarmBlooded (Vipera)
· Human (Diego) · Animal (Vipera) messa per la frase "there are animals that are not warmblooded"
· Human (Sofia) · Mammal (Macaco)
· Own (Carlo, Fuffy) · Own (Sofia, Vipera)

FORSE

• I think that before defining individuals (before) and axioms (later) we need to express a language. ("Assume and encode further elements...")
· Human (x)
· Mammal (x)
· WarmBlooded (x) (¬WarmBlooded(x))
· Dog (x)
· Own (x,y) binary predicate to express that x owns y.
· Animal (x)

3) Write an encoding of the KB using the chosen KRL.

$\forall x.(Human(x) \rightarrow Mammal(x))$

$\forall x.(Mammal(x) \rightarrow WarmBlooded(x))$

$\forall x.(\neg Animal(x) \land Dog(x) \rightarrow Mammal(x))$

$\exists x \exists y.(Human(x) \land Animal(y) \land own(x,y))$

↳ non ci va probabilmente perché la frase è scritta senza quantificatori

$\exists x (Animal(x) \land \neg WarmBlooded(x))$ la vipera ad esempio

$\forall x (Mammal(x) \rightarrow Animal(x))$

SECONDO ELEONORA NON SERVE AGGIUNGERE QUESTO FORMALISMO PERCHÉ SCRIVIAMO GIÀ GLI INDIVIDUALS

$\neg \exists x \exists y (Human(x) \wedge Human(y) \wedge \neg(x=y) \wedge Own(x,y))$

Plus all the facts enumerated in point 2 plus the facts determining the language (I think)

1) We can query our KB using for example Prolog and so we get:

?- bagof((x,y), (Human(x), Animal(y), WarmBlood(y), Own(x,y)), S).

?- findall(x, (Human(x), WarmBlood(y), WarmBlood(z)), S)

③

1) We have a NON-STATIC DOMAIN therefore it's convenient to use SITUATION CALCULUS which allows us to reason on actions and resulting states.
(we need to produce a sequence of actions that allows the robot to perform its objective → a PLAN → planning in situation calculus with situation varying over time).

2) • A little cleaning robot moves in a room

robot → the agent

moves → goes from one position to another

goes → ACTION

position → positions are encoded with coordinates x, y → the STATES represented by fluent "At"

• A cat and a Dog are also in the room and they do not move → this condition is ATEMPORAL

→ they are AT same position and they always stay there.

• Robot should clean the room (visit all the free squares) avoiding the cat and the dog.

→ what we understand → clean is an action
(but it could also be a state (soluzione Eleonora - vedi dopo -) and so a fluent, if we consider that just going to a position, the robot automatically clean the space, so clean in that case is something that can change as an effect of the action go
(1 clean before going there, clean after going there)

• I have to encode the concept of free.

• Inner concept: non c'è scritto esplicitamente ma ovviamente devo codificare che esso può muovere solo tra celle adiacenti → inserirò un predicato "Adjacent" Atemporal

SOLUZIONE PROF

- FLUENTS:
- At(Agent, cl, place, s) # cl represents the list of cells to be cleaned
  # place is a pair constructed with coordinates (x,y)

- ATEMPORAL PREDICATES:
  (what remains the same)
  At(cat,[x,y]) (rimane sempre lì dov'è specificato nell'initial state)
  At(dog,[x,y])
  Adjacent([x,y],[x',y']) (stessa cosa per le adiacenze)
  Free(x) $\longleftrightarrow$ ¬ At(cat, x) ∧ ¬ At(dog, x)
- INITIAL STATE:
  At(robot, [[1,2],[1,3],[2,1] ...], [1,1], S0) (1) → AND
  initial position of the robot

  At(cat,[2,3]),
  At(dog,[3,3]),
  Adjacent([1,1],[1,2]),
  Adjacent([1,1],[2,1]),
  Adjacent([2,1],[3,1]),
  Adjacent([2,1],[2,2]),
  Adjacent([1,2],[2,2]),
  Adjacent([1,2],[1,3]),
  Adjacent([2,2],[2,3]),
  Adjacent([2,2],[3,2]),
  Adjacent([3,3],[2,3]),
  Adjacent([3,3],[3,2]),
  (sono free tutte le caselle dove non sono presenti gatto o cane, andrebbe scritto)
  (troppo lungo)

- ACTIONS:

  Go(x, y) sarebbe Go(x^1, y, x', y') visto che abbiamo coordinate

  Clean(x, cl)

  Finish(x) to express that robot has finished in position x. (x, y)

- POSSIBILITY AXIOMS

  GO: posso andare in y se mi trovo in x, se x e y sono adiacenti e se y non è occupato dal cane o dal gatto.

  → 2 POSSIBILITY AXIOMS perché posso andare da x a y se x è adiacente a y, ma anche se y è adiacente a x

At(robot, cl, x, s) ∧ Adjacent(x, y) ∧ Free(y) → Poss(Go(x, y), s)

At(robot, cl, x, s) ∧ Adjacent(y, x) ∧ Free(y) → Poss(go(x, y), s)

At(robot, cl, x, s) ∧ member(x, cl) → Poss(clean(x, cl), s)

? questo predicato non è espresso da nessuna parte, come viene considerato? atemporal or fluent?

At(robot, [], x, s) → Poss(finish(x), s)

- EFFECT AXIOMS:

Poss(go(x, y), s) → At(robot, cl, y, Result(go(x, y), s))

Poss(clean(x, cl), s) → At(robot, remove(x, cl), x, Result(clean(x, cl), s))

? come member

Poss(Finish(x), s) → ~~At(robot~~ Finish(x, Result(finish(x), s))

IO AVREI DETTO:

At(robot, [], x, Result(Finish(x), s))

GOAL:

∃ seq Finish(x, Result(seq, s0))

E il frame problem? come viene risolto?

④

1) To formalize the sentences we can use FOL because we do not need the expressivity of DL or SC.

Formalization:

- F1. $\forall x.(Student(x) \to Smart(x))$
- F2. $\exists x.Student(x)$
- F3. $\exists x.(Student(x) \land Smart(x))$
- F4. $\forall x(Student(x) \to \exists y(Student(y) \land loves(x,y)))$
- F5. $\forall x(Student(x) \to \exists y.(Student(y) \land loves(x,y) \land \neg(x=y)))$
- F6. $\exists x.(Student(x) \land \forall.y(Student(y) \land \neg(x=y) \to loves(y,x)))$
- F7. $Student(Mark)$
- F8. $Student(Paul)$
- F9. $(Takes(Mark, Analysis) \land \neg Takes(Mark, geometry)) \lor$
  $(\neg Takes(Mark, Analysis) \land Takes(Mark, geometry))$
- F.10 $Takes(Paul, Analysis) \land Takes(Paul, geometry)$
- F11. $\neg Takes(Mark, Analysis)$
- F12 $\neg\exists x.(Student(x) \land loves(x, Paul))$

2) The KB is CONSISTENT, since it exists an interpretation that (che comporta) this.

PROOF:

$D = \{Mary, Lisa\}$ 2 possibili persone - DOMAIN -

Proved sentences

- By definition F7 and F8 are true in every interpretations (they are facts)
- Moreover F9, F10, F11 are true in any interpretation.

→ all ground variables

- By defining:

$loves(Mary, Paul) = \bot$ (false)
$loves(Mark, Paul) = \bot$
$loves(Lisa, Paul) = \bot$

we get $F12 = T$

• from $F_8 = T$ (student(Mark)) we get also $F_2 = T$

• We make $F_6 = T$ by defining:

loves(Paul, Mark)
loves(Lisa, Mark)
loves(Mary, Mark)

• let's define

Smart(Mark)
Smart(Paul)
Smart(Lisa)
Smart(Mary)

and we get $F_1 = T$ and consequently $F_3 = T$

• By adding:

loves(Mark, Lisa)

we get also F_4 and F_5 true

Therefore $F_1 \land F_2 \land F_3 \land F_4 \land F_5 \land F_6 \land F_7 \land F_8 \land F_9 \land F_{10} \land F_{11} \land F_{12} = T$ under the given interpretation and the KB is consistent.

3) I decide to use Prolog to write my query.

```
?- takes(Paul, X).
```

or:
```
findall(X, (takes(Paul, X)), S).
```
S will be the list containing all the exams taken by Paul.

5

1) To formalize the sentences we can use FOL because we do not need the expressivity of DL or SC.

2) when we formalize with FOL we define a language (predicates that we will use) and a set of axioms using the specified language

LANGUAGE:

My solution:

- carpool(Driver, Passenger) *
- Day(x) x is a day
- Odd(x) x is odd (I don't specify "Even" predicate because I can express it as the negation of odd)
- hasDriveLicense(Person)
- Ill(Person)

Professor solution: carpool(Driver, Passenger, Place, Day)

* I assume that the only possible carpooling can be done in order to go to work.

- Formalization:

- $\forall x.\ Day(x) \land \neg Odd(x) \land hasDriveLicense(Clara) \rightarrow carpool(Clara, Doris)$
- $\forall x.\ Day(x) \land Odd(x) \land hasDriverLicense(Doris)) \rightarrow carpool(Doris, Clara)$
- $\forall x\ Day(x) \land Ill(Doris) \land hasDriveLicense(Clara) \rightarrow carpool(Clara Doris)$
- $hasDriveLicense(Clara)$
- $hasDriveLicense(Doris)$

we assume for each working day the facts:

$odd(day1)$ $\neg odd(day2)$
$odd(day3)$ $\neg odd(day4)$
$odd(day5)$
$odd(day7)$

DIVERSO DA SOLUZIONE PROF

Extension of KB:

3) $\neg Odd(current\_day)$
$\neg hasDriveLicense(Clara)$

If we extend the KB with $\neg hasDriveLicense(Clara)$, it become inconsistent and it's not possible to prove that they are able to carpool to work.

To represent this extension a default logic is needed and an additional condition should be added and formalized →