Knowledge Representation and Reasoning

Jef Wijsen

UMONS

January 30, 2024

Course Content

$$\text{AI} \simeq \text{GOFAI} + \text{AML}$$

- AI: Artificial Intelligence
- GOFAI: Good Old-Fashioned Artificial Intelligence ($\simeq$ symbolic AI)
- AML: Adaptive Machine Learning (reinforcement learning, big data...)

$$\text{KRR} \subseteq \text{GOFAI}$$

- KRR: Knowledge Representation and [Automated] Reasoning

Course Content

Logic-based KRR

- Answer Set Programming (ASP): an expressive logic for specifying and solving problems in NP (including NP-complete problems).

  Task: Solving
  Input: A set Σ of logic formulas (also called constraints, rules...). (e.g., the rules of Sudoku + a partially filled grid)
  Question: Is there a solution (also called model, answer set...) that satisfies every formula in Σ?

- Web Ontology Language (OWL): less expressive logics that allow automated reasoning about data on the Web.

  Task: Automated reasoning
  Input: A set of logic formulas Σ; a logic formula σ.
  Question: Is σ a logical consequence of Σ?

Note: Automated reasoning is computationally impossible for expressive logics.

Recall from Bases de Données I and II

$$\text{FO} \subsetneq \text{Datalog}^{\neg} \subseteq \text{P} \subseteq \text{NP} \subseteq \text{Prolog}$$

where

- FO denotes the class of problems that take as input a relational database instance and can be solved by a query in relational calculus; and
- Datalog$^{\neg}$ denotes the class of problems that take as input a relational database instance and can be solved by a program in Datalog with stratified negation.

Recall:

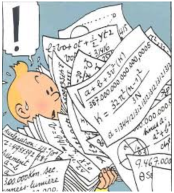
- NP-complete problems cannot be programmed in Datalog$^{\neg}$.
- Automated reasoning is already computationally impossible for FO.

Datalog$^{\neg}$

See "A Datalog Primer."

http://informatique.umons.ac.be/ssi/teaching/bdIImons/primerDatalog.pdf

Course Methodology

	Classical course	⇝	This course
language	French	⇝	English
teacher's role	teaching	⇝	guiding
students' role	   being taught	⇝	scientific discovery tour
evaluation	exam	⇝	project + homeworks + written exam

Course Schedule

Just a screenshot (the full schedule is online):

This document may be updated during the course.

Tuesday, Feb. 6 (15H45)	Meeting in room P.3E11 + organization (14')
Wednesday, Feb. 7 (15H45)	motivation (72')
Thursday, Feb. 8 (15H45)	introduction (170')
Friday, Feb. 9 (10H30)	Meeting in P.0A07; start Homework 1 (due on Feb. 22)Y
Tuesday, Feb. 13 (15H45)	modeling (106')
Wednesday, Feb. 14 (15H45)	YC
Tuesday, Feb. 20 (15H45)	Meeting in B4.233; start Homework 2 (due on Mar. 4)
Thursday, Feb. 22 (15H45)	language (128')
Tuesday, Feb. 27 (15H45)	
Wednesday, Feb. 28 (15H45)	Meeting in B4.233; start Homework 3 (due on Mar. 12)
Thursday, Feb. 29 (15H45)	
Friday, Mar. 1 (10H30)	

Datalog$^{\neg}$ by Example

```
red(a,b). red(b,c). red(c,a).
blue(a,c). blue(c,d). blue(d,a).
redTrans(X,Y) :- red(X,Y).
redTrans(X,Z) :- redTrans(X,Y), red(Y,Z).
blueMonopoly(X,Y) :- blue(X,Y), not redTrans(X,Y).
```

- `redTrans` and `blueMonopoly` are IDB predicates (because they occur in rule heads); the other predicates are EDB predicates (= stored database relations).
- The PDG (Program Dependence Graph) has a (non-negated) edge from `redTrans` to `redTrans`, and a negated edge from `blueMonopoly` to `redTrans`.
- Stratified semantics: execute the rules for `redTrans` until no more `redTrans`-facts can be derived; only then can rules with "`not redTrans`" be evaluated.

ASP by Example

```
person(john).
happy(X) :- person(X), not unhappy(X).
unhappy(X) :- person(X), not happy(X).
```

Not stratified: the 1st rule should be executed before the 2nd rule (because of "`not happy`"), but the 2nd should be executed before the 1st (because of "`not unhappy`").

An ASP solver will find two models:

```
clingo version 4.5.4
Solving...
Answer: 1
person(john) happy(john)
Answer: 2
person(john) unhappy(john)
SATISFIABLE

Models       : 2
```