# INF5390
# Obligatorisk oppgave 2

Magnus Andersen

April 21, 2013

## 1 First-order logic

We are going to represent the following sentences using first-order predicate logic:

a. Not all students are enrolled in both History and Biology.

b. Only one student failed History.

c. Only one student failed both History and Biology.

d. The best grade in History is better than the best grade in Biology.

In order to do this, we will use the following consistent vocabulary, consisting of constants and predicates:

- Constants:
  - *History* : a course in which students enroll
  - *Biology* : a course in which students enroll
- Predicates:
  - $Student(s)$ : True if $s$ is a student
  - $Enrolled(s,\ c)$ : True if $s$ is enrolled in course $c$
  - $Failed(s,\ c)$ : True if $s$ failed course $c$
  - $BestGrade(g,\ c)$ : True if $g$ is the best grade in course $c$
  - $Better(g_1, g_2)$ : True if grade $g_1$is better than grade $g_2$

Using the above vocabulary, we can represent a.-d. in first-order logic:

a. "Not all Xs do Y" is the same as saying "At least one X does *not* do Y". Therefore: $\exists s(Student(s) \land \neg(Enrolled(s, History) \land Enrolled(s, Biology)))$

b. $\exists! s(Student(s) \land Failed(s, History))$

c. $\exists! s(Student(s) \land Failed(s, History) \land Failed(s, Biology))$

d. $\exists g_1(BestGrade(g_1, History) \land \exists g_2(BestGrade(g_2, Biology) \land Better(g_1, g_2)))$

where $\exists!$ is the uniqueness quantifier (i.e. "there exists one and only one").

## 2 Agents that plan

Here our task is to define action schemas for the problem of putting on socks and shoes, hat and coat. We shall then define and sketch a partially ordered plan that solves the problem, and consequently evaluate the number of different linearizations of the solution.

The action schemas are:

| | |
|---|---|
| $\boldsymbol{Action}(PutOn\text{Socks})$,<br>PRECOND: N/A<br>EFFECT: $SocksOn$ | $\boldsymbol{Action}(PutOn\text{Shoes})$,<br>PRECOND: $SocksOn$<br>EFFECT: $ShoesOn$ |
| $\boldsymbol{Action}(PutOnHat)$,<br>PRECOND: N/A<br>EFFECT: $HatOn$ | $\boldsymbol{Action}(PutOnCoat)$,<br>PRECOND: N/A<br>EFFECT: $CoatOn$ |

Table 1: Action schemas

The partially ordered plan can be defined as follows:

**Initial state**:

- $\neg SocksOn \wedge \neg ShoesOn \neg HatOn \wedge \neg CoatOn$ (== *Undressed*)

**Goal state:**

- $SocksOn \wedge ShoesOn \wedge HatOn \wedge CoatOn$ (== *Dressed*)
- **Actions:**
  - *PutOnSocks*, PRECOND: N/A, EFFECT: $SocksOn$
  - *PutOnShoes*, PRECOND: *SocksOn*, EFFECT: $ShoesOn$
  - *PutOnHat*, PRECOND: N/A, EFFECT: $HatOn$
  - *PutOnCoat*, PRECOND: N/A, EFFECT: $CoatOn$

A sketch of a partial order plan is shown in Figure 1.

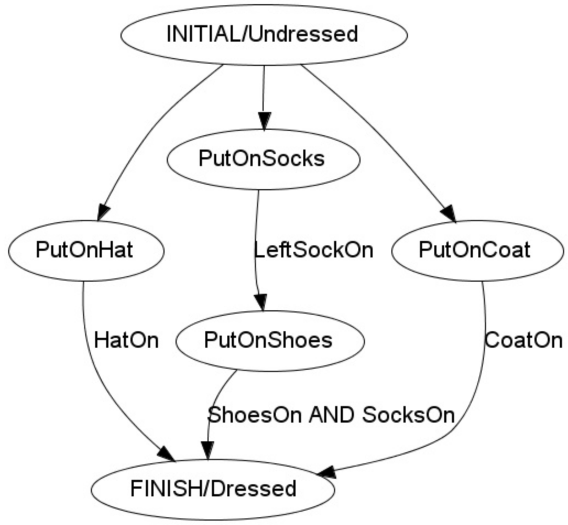


Figure 1: A partial order plan

If the order in which one equips the different clothes did not matter at all (socks/shoes included), the number of different permutations/orderings/linearizations would be $4! = 24$ (4 initial positions; one for each piece of clothing). But since the socks need to be equipped *before* the shoes, this is number would obviously be erroneous. Specifically, this will slice the previous number in half, to 12 linearizations:

- Starting with socks: 3 free (order unimportant) actions (hat/shoes/coat) to choose from, thus $3! = 6$ permutations.
- Starting with hat: 2 free actions (coat/socks) to choose from, but have to add 1 since the shoes are "locked" to the socks; $2! + 1 = 3$ permutations.
- Starting with coat: 2 free actions (hat/socks) to choose from, but have to add 1 as per the previous bullet point; $2! + 1 = 3$ permutations.
- So in total: $6 + 3 + 3 = 12$ permutations/linearizations.

(The number of action orderings would be vastly greater if we specified the problem with left/right socks and shoes, but similar reasoning could be done).

# 3 Agents that reason under uncertainty

A factory alarm goes off if a thermometer reading exceeds a certain threshold value. We define the Boolean variables $A$ (alarm produces sound), $F_A$(alarm fails) and $F_M$(thermometer fails), along with the multivalued variables $M$ (thermometer) and $T$ (actual temperature).

## a) Bayesian network

The thermometer has a higher probability of failing when the temperature gets too high. With this in mind, we are going to sketch a Bayesian network of using the domain variables specified above. The dependencies needed are:

- As per the above, the probability of a faulty thermometer increases with actual temperature. Thus $T \to F_M$.
- The thermometer reading is obviously dependent upon the actual temperature. Thus $T \to M$.
- The event in which the thermometer fails obviously has an impact on the thermometer reading. Thus $F_M \to M$.
- Since the alarm goes off if the thermometer reading exceeds some (unspecified) limit, the alarm is obviously dependent upon the thermometer reading. Thus $M \to A$.
- Finally, the event in which the alarm goes off is dependent upon whether the alarm fails. Thus $F_A \to A$.

The network is illustrated in Figure 2.

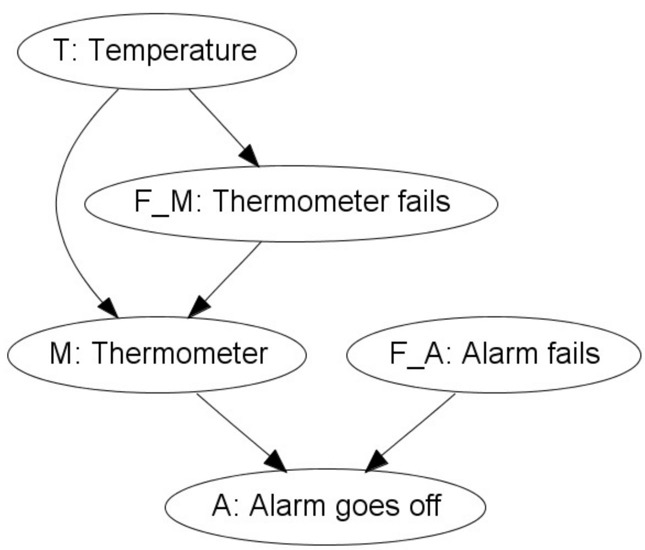


Figure 2: Bayesian network

## b) Conditional Probability Table (CPT) for the alarm

We see from Figure 2 that the alarm, $A$, is dependent upon the thermometer reading, $M$, and the event in which the alarm fails, $F_A$.

We assume that there are only two possible thermometer readings: *Normal* and *High*. That is, $M$ can now be thought of as a Boolean variable, with e.g. False/True indicating Normal/High temperature, respectively. We also assume that the alarm is always in working order unless it has failed (== not producing sound). In other words: if we assume that the alarm is supposed to go off whenever the thermometer reading is High (== exceeded threshold value), it will always go off when the temperature is High if it is not faulty ($\neg F_A$).

Based on these assumptions, we can define the conditional probability table (CPT) for $A$ as shown in Table 2.

| $M$: Temperature reading | $F_A$: Alarm fails | $P(A)$ | $P(\neg A)$ |
|---|---|---|---|
| False | False | 0 | 1 |
| False | True | 0 | 1 |
| True | False | 1 | 0 |
| True | True | 0 | 1 |

Table 2: CPT for the alarm