TUTORIAL 3 · [COMPUTATION AND LOGIC]

OBJECTIVES

In the first part of this tutorial, you will:

– work with *syllogisms* and meet Lewis Carroll, the logician.

In the second part you will:

– learn more about *sequents*, *satisfaction*, and *operations with predicates*;

– challenge the expressivity of *Aristotle's categorical propositions*.

TASKS

Exercises 1 and 4–6 are mandatory. Exercises 2, 3 and 7 are optional.

SUBMIT a file called `cl-tutorial-3` with your answers that do not require Haskell code (image or pdf) and the file `Things.hs` with your code.

DEADLINE Saturday, 10th of October, 4 PM UK time

Good Scholarly Practice

Please remember the good scholarly practice requirements of the University regarding work for credit.

You can find guidance at the School page

http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct.

This also has links to the relevant University pages. Please do not publish solutions to these exercises on the internet or elsewhere, to avoid others copying your solutions.

PART A

MANDATORY | BEFORE TUTORIAL SESSION

Read Chapters 7 (*Patterns of Reasoning*) and 8 (*More Patterns of Reasoning*) of the textbook.

Lewis Carroll, author of *Alice's Adventures in Wonderland* and *Through the Looking Glass*, taught logic at Oxford in the 1890s. In his words:

> *"When the terms of a proposition are represented by words, it is said to be in concrete form; when by letters, abstract."*

To translate a proposition from concrete to abstract form, we choose a universe and regard each term as a predicate, to which we assign a letter.

He gives the following example:

	All cats understand French	$a \vDash b$
	Some chickens are cats.	$c \nvDash \neg a$
$\therefore$	Some chickens understand French.	$c \nvDash \neg b$

where *universe* = animals, a = cats, b = understanding French, c = chickens.

We can put this (sound) argument into inference form:

$$\frac{a \vDash b \qquad c \nvDash \neg a}{c \nvDash \neg b}$$

EXERCISE 1

Consider the following two arguments by Lewis Carroll:

All diligent students are successful.	Every eagle can fly.
All ignorant students are unsuccessful.	Some pigs cannot fly.
∴ Some diligent students are ignorant.	∴ Some pigs are not eagles.

Formulate them as syllogisms, then use Venn diagrams to check whether they are sound, i.e., whether they are correct arguments or not.

For each of the arguments, if it is sound, give a proof involving Venn diagrams, then derive it from Barbara using denying the conclusion, substituting for predicates, contraposition and the double negation law. If the argument is not sound, then give a counterexample.

EXERCISE 2

OPTIONAL | BEFORE TUTORIAL SESSION

What can you say about the number of occurrences of $\not\models$ in any sound syllogism? How about the number of occurrences of $\neg$?

Consider the following syllogism:

$\quad$ No animals are unicorns.
$\quad$ All unicorns are horses.
$\therefore$ Some horses are not animals.

Translate it to symbolic form and explain, using the answers given to the two questions above, without further calculation, and without giving a counterexample, why it is not sound.

EXERCISE 3

In Chapter 8 of the textbook, at page 66, you saw the list of all valid syllogisms. In total, the list contains 15 syllogisms.

Can you convince yourself that this list is indeed exhaustive? That is, there is no other valid syllogism which is not on that list.

This is a challenging question: kudos if you solve it; if you don't, no worries, it is safe to assume it is correct for the following exercise.

EXERCISE 3 (CONT.)

OPTIONAL | BEFORE TUTORIAL SESSION

Aristotle considered 9 other syllogisms to be sound. This is not because he made a mistake, but because he made an assumption that we do not make: under the so-called *existential assumption*, $a \vDash b$ also requires that there is at least one thing that satisfies a.

To derive Aristotle's 9 extra correct arguments, it suffices to add the existential assumption as a rule with no premise of the form:

$$\frac{}{\text{“some } a \text{ is } a\text{”}}$$

Convince yourself that the existential assumption can be expressed as “some a is a”. Then write the conclusion of the rule above as a sequent.

EXERCISE 3 (CONT.)

OPTIONAL | BEFORE TUTORIAL SESSION

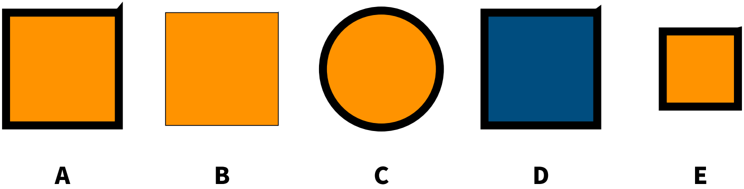
Check that the following syllogism by Aristotle is correct under the existential assumption using Venn diagrams.

$$\frac{a \vDash b \qquad c \vDash a}{c \nvDash \neg b}$$

Derive the syllogism from the 15 correct arguments in the textbook (page 66) and the *existential assumption rule* given on the previous slide.

PART B

In this tutorial, we'll be reusing the universe of 5 things from our previous tutorial:



⚠ The file `Things.hs` contains a description in Haskell of this universe. You will use it as a basis for your solutions to exercises that require Haskell code.

EXERCISE 4

MANDATORY | BEFORE TUTORIAL SESSION

Read the subsection on *Sequents* from Chapter 6 (*Features and Predicates*) of the textbook, on page 53.

Express each of the following using a sequent:

1. “Every big amber thing has a thick border” is false.
2. “Some small thing is a disc” is true.
3. “Some small square is amber” is false.

EXERCISE 5

MANDATORY | BEFORE TUTORIAL SESSION

Define (and add to the file `Things.hs`) an infix function:

```
(|=) :: Predicate Thing -> Predicate Thing -> Bool
```

for testing whether a sequent involving one antecedent and one succedent is true or false. Use it to check that `isDisc` ⊨ `isAmber`.

Now implement an infix function:

```
(|/=) :: Predicate Thing -> Predicate Thing -> Bool
```

so that `a |/= b` is true if and only if some `a` is not `b`.

EXERCISE 5 (CONT.)

Define an infix function:

```
(||=) :: [Predicate Thing] -> Predicate Thing -> Bool
```

for testing whether a sequent involving a list of antecedents and one succedent is true or false. Use it to check that:

- `isBlue`, `isSquare` ⊨ `isBig`
- `isBig`, `isAmber` ⊭ `isDisc`

EXERCISE 6

Recall that the type `Predicate u` is defined as `u -> Bool`. The following function negates a predicate:

```
neg :: Predicate u -> Predicate u
(neg a) x = not (a x)
```

For example, `(neg isAmber) C = not (isAmber C) = False`, and `isBlue |= neg isSmall` produces `True`.

Define functions

```
(|:|) :: Predicate u -> Predicate u -> Predicate u
(&:&) :: Predicate u -> Predicate u -> Predicate u
```

that compute the disjunction and conjunction of two predicates.

EXERCISE 6 (CONT.)

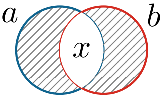
MANDATORY | BEFORE TUTORIAL SESSION

Which of the following produce `True`?

1. `isBig &:& isAmber |= isDisc`
2. `isBig &:& isDisc |= isAmber`
3. `isSmall &:& neg isBlue |= neg isDisc`
4. `isBig |:| isAmber |= neg isSquare`
5. `neg (isSquare |:| isBlue) |= hasThickBorder`
6. `neg isSquare &:& neg isAmber |= isDisc`

Revisit section *Venn diagrams with inhabited regions* from Chapter 8 of the textbook, at page 64. We'll work now with diagrams whose regions are either empty (hatched) or inhabited (marked with an x, y, or z, etc.).

Given two predicates a and b, any Venn diagram with two circles (one for a, one for b) is uniquely determined by the set of *all* Aristotelian propositions ("Every a is b", "No a is b", "Some a is b", or "Some a is not b") that are valid for that diagram. For example, the Venn diagram



is determined by $\{$Every a is b, Every b is a, Some a is b, Some b is $a\}$. It's the only diagram with two circles for which these propositions are true.

Does this property still hold for three predicates (and Venn diagrams with three circles, one for each predicate)?