Artificial Intelligence

Exercise sheet 1
Introduction, Propositional Logic & Predicate Logic

1 Introduction

Note: This is an individual task, you should answer it with your own words.

Intelligent Systems (8 points)

Read a scientific article about the past and future of the "Artificial Intelligence". Use the last digit of your Registration Number (Matrikelnummer) to identify the article assigned to you.

0. Man versus Machine or Man + Machine? - M.M. Cummings (2014), `http://dx.doi.org/10.1109/MIS.2014.87`
1. Artificial Intelligence and Big Data - D.E. O'Leary (2013), `http://dx.doi.org/10.1109/MIS.2013.39`
2. AI's War on Manipulation: Are We Winning? - P. Faliszewski & A.D. Procaccia (2010), `http://dx.doi.org/10.1609/aimag.v31i4.2314`
3. Human-Level Artificial Intelligence? Be Serious! - N.J. Nilsson (2005), `http://dx.doi.org/10.1609/aimag.v26i4.1850`

4. Where's the AI? - R.C. Schank (1991),
   http://dx.doi.org/10.1609/aimag.v12i4.917
5. The Knowledge Level - A. Newell (1981),
   http://dx.doi.org/10.1609/aimag.v2i2.99
6. The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering - E.A. Feigenbaum (1977),
   http://www.ijcai.org/Past%20Proceedings/IJCAI-77-VOL2/PDF/092.pdf
7. ELIZA–a computer program for the study of natural language communication between man and machine - J. Weizenbaum (1966),
   http://dx.doi.org/10.1145/365153.365168
8. Programs with Common Sense - J. McCarthy (1959),
   http://www-formal.stanford.edu/jmc/mcc59.pdf
9. Computing Machinery and Intelligence - A.M. Turing (1950),
   http://www.cs.umbc.edu/471/papers/turing.pdf

Task: prepare some slides

- Contents: A **summary of the highlight** of the article
- Number of pages: 5-7, in the format of Powerpoint or PDF

2 Propositional Logic

> *Note: This is a group task, you may solve it in groups up to the size of two people, everybody must however be able to explain all parts of the implementation*

You'll have to implement an object model for: (i) the syntax, (ii) an interpretation for the semantics, (iii) an algorithm for reasoning. You may use the programming language of your choice, but Java [1] and Eclipse [2] are recommended programming language and development environment respectively.

2.1 Syntax (6 points)

Create an object model [3] that is able **to represent the syntax of propositional logic**. You do not need to implement a parser (the object model can be cre-

[1] Java, https://www.java.com/
[2] Eclipse, http://www.eclipse.org/
[3] e.g. a Java class hierarchy that is capable of representing every possible sentence in propositional logic in Java objects

ated programmatically), but the object model must be serializable [4] (you can use textual representation, i.e. "and" representing $\wedge$).

Test your implementation by representing a sentence like e.g. $A = (\neg p \rightarrow q) \leftrightarrow (p \vee q)$ and test your serialization functionality.

Exercise 2 - Semantics (8 points)

Write a function **to apply an interpretation** [5] to a sentence represented in your object model. Furthermore, your implementation should be able to compute the **truth value** of that sentence after assigning the interpretation.

Let v is an assignment function, test your implementation on formula 2.1.
If $v(p) = F$ and $v(q) = T$, determine what is $v(A)$?

$$A = (\neg p \rightarrow q) \leftrightarrow (p \vee q) \tag{2.1}$$

Can you use your implementation to test whether your sentence is a tautology?

2.2 Reasoning (8 points)

Implement **propositional resolution** for your object model. You do not need to implement an algorithm to transform arbitrary expressions into clausal form, you need to test it only on expressions in CNF.

Use your implementation to test if formula 2.2 is a valid formula.

$$(p \rightarrow q) \rightarrow (\neg q \rightarrow \neg p) \tag{2.2}$$

3 Predicate Logic

> *Note: This is a group task, you may solve it in groups up to the size of two people, everybody must however be able to explain all parts of the implementation*

You'll have to extend/adapt the object model from the last exercise (propositional logic).

[4] i.e. serialized to a textual representation on the console

[5] the interpretation could be represented by `Map<Proposition, Boolean>`, i.e. when applied to a sentence it applies a truth value to all propositions in the sentence

3.1 Syntax (9 points)

Extend the object model from the exercise of propositional logic **to capture the syntax of predicate logic**. Again, you do not need to implement a parser (the object model can be created programmatically), but the object model must be serializable (you can use textual representation, i.e. "and" representing $\wedge$).

Test your implementation by representing a formula containing **at least one function symbol** and **one quantification**.