Report: Knowledge Representation and Reasoning (KRR) Assignment

CS19M012: Ashish Gupta

CS19M027: Hemant Poharkar

Problem Statement:

Read a set of formulas from a text file in FOL, convert it to XML and produce its Implicit Quantified form. Implement the Unification algorithm to operate on the XML notation. Implement a Forward Chaining algorithm to accept a KB in the specified form and generate a proof for a given theorem/query. The proof should be in the form of a numbered sequence of statements starting with the statements from the KB used in the proof.

Solution Approach:

The problem statement has 3 inherent subtasks which was supposed to be solved. First task was to perform skolemization on a set of formulas, 2nd task was to implement unification and 3rd task was to implement forward chaining algorithm using the unification module.

1st Task: Skolemization

As explained in the lecture by sir, we have 3 cases and an exception case to handle given the XML generated by KRR toolkit. In the first case, we replace all universal quantifiers as follows: $\forall x$ is converted to $?x$. There are 2 straight cases pertaining to existential quantifiers. In the first case, we do not have any universal quantifier before ∃. So, we replace existential quantifier variable with a skolem constant (SK followed by a variable name). In the 2nd case, we have universal quantifiers before ∃. For such cases, we replace the existential quantifier variable with skolem function with parameters as universally quantified variables. For the exception case, we have existential variable in the left side of the implication. So, for such cases, we treat the existential variable as universal variable. After all the replacement, we write the XML to a new file.

2nd Task: Unification

We implement unification to make 2 different FOL atomic formulas identical by finding a substitution. It returns the most general unifier (MGU) as the output of the unification algorithm. We have implemented the algorithm shared by sir in the class. We have taken care for the following basic conditions like number of arguments being identical, predicate symbol should be the same. We have implemented few helper methods like `occur_check` which determines that we if a variable exists in a substitution list and `get_substitution_value` which returns the substitution value for a given value. These helper methods help us specifically in the failure condition of unification.

3rd Task: Forward Chaining

For forward chaining, we have 3 text files viz. fchain-kb.txt, fchain-query.txt and rules.txt. We first convert them into XML using the toolkit provided. Then we load the KB in a list. We have considered only Modus Ponens rule to implement forward chaining. We take two predicates and unify them using the unification algorithm implemented above. Once unified, we substitute the value and match it with the rule. If the rule is resolved, the consequent is checked with query mentioned in the query file and if it matches, we stop the algorithm else we add it to the KB and take next set of formulas and repeat the procedure.

Tools used:

Language: Python

Library/API: xml.etree.ElementTree API, KRR toolkit

Issues/Challenges:

For forward chaining, we were able to implement only rule of inference i.e Modus Ponens.

Contributions:

CS19M027 (Hemant Poharkar): Skolemization

CS19M012 (Ashish Gupta): Unification, Forward Chaining