# CS421 Unit Project Proposal – Implementation of Prolog in Ocaml

He Xiao, Shijiao Yuwen

November 16, 2014

## Brief Background

Prolog is a logic programming language which uses unification algorithms intensively. Given a query, Prolog will try to unify the query with the existing facts up to some term rewritings. If there exists some substitution function that can satisfy all the constraints during unification, then Prolog will answer yes and give the first substitution function as solution. If the user wants to see more solutions, then Prolog will use some backtracking techniques to try to generate the subsequent substitution function as output. If there is no solution left, then Prolog simply answers no.

## Aim

Implement a subset of Prolog in Ocaml. Delivered product should include but not limited to lexer, grammar, parser, interpreter as well as sufficient test cases and user manual.

## Design idea

- Define the syntax and grammar of language 'Prolog-Ocaml'.
- Transform the source code of prolog prgram (read from file or user's command line input) to the AST in ocaml which will be used by unification and resolution. At current stage, we believe that a list of pairs of a single term and a horn clause will suffice (representing the list of head-body mappings in pure prolog program). A fact 'c' in prolog will be transformed to pair (c, true).
- When running the interpreter, structural induction will be performed on the initial query; based on the structure, either result is found, or subgoal is generated and the recursion function is called, or the initial query is decomposed to several sub-goals and the final result will be based on the combination of the results of these sub-goals.

## Plan of actions

### Week 1: 10 Nov - 16 Nov

Background reading and investigating deeply the MP7 (Unification algorithm). Try a toy example of unification which involves application of a simple rule (With the data structures hard-coded in Ocaml).

### Week 2: 17 Nov - 23 Nov

Continue reading some materials about horn clauses. Write algorithm that transforms the logical expressions to horn clauses. Try a larger real prolog program and translate it by hand to data structures in Ocaml and test the unification engine works.

### Week 3: 24 Nov - 30 Nov

Define the syntax (regular expression) and BNF grammar of the simplified Prolog. Implement the lexer and parser such that given a source program written in the acceptable syntax, the internal Ocaml data structure (AST) will be generated and used by the interpreter to evaluate and return the resulting substitution function if it exists.

### Week 4: 1 Dec - 7 Dec

Complete the whole system. Optimize the program and prepare the demo and report.

## Teamwork

Both He Xiao and Shijiao Yuwen will read the papers about Prolog and unification algorithms. Shijiao is responsible for the coding of lexer and parser, He is responsible for designing the unification algorithm, resolution algorithm (with backtracking), and evaluation algorithm. After any code snippet is written, it will be reviewed by the other member.