Assignment - In progress

Complete the form, then choose the appropriate button at the bottom.

Title	Assignment 5: Simple Prolog interpreter
Due	7 Apr, 2014 11:55 pm
Status	Not Started
Grade Scale	Points (max 10.0)

Instructions

In this assignment, you wil code a complete symbolic Prolog interpreter.

You need to have iplemented unification efficiently, and backtracking, using a stack. You also need to maintain lists of subgoals.

A Prolog program consists of a sequence of Horn Clauses (variables in each are implicitly universally quantified).
A Horn Clause is either a Rule or a Fact.
All clauses have a "head" and are terminated by a dot "."
Facts only have a head, which is an Atom.
Rules have a head, then the connector (read as "if") ":-" followed by a "body".
The body is a sequence of Atoms, spearated by commas "," and terminated by a dot "."
An atom consists of a predicate symbol followed optionally by a list of terms enclosed in parentheses (and) and spearated by commas ",".
A predicate symbol is an alphanumeric identifier beginning with a small letter. Special characters underscore "_" and prime (') may also be used.
A term can either be a Variable (an alphanumeric identifier beginning with a Capital letter); or a term can be an Atom.

A goal clause is a sequence of atoms, separated by commas.

1. You need to implement in OCaml a parser and tokenizer for this language. Write a suitable grammar that is unambiguous and can be processed by Yacc
2. Design a suitable intermediate abstract syntax.
3. Implement a symbol table using which you can check the arities and well-formedness of terms and atoms
4. You need to write an interpreter which can load a given program and then interactively take a goal and execute
it, giving answers as a Prolog interpreter does.

Submission

Assignment Text

This assignment allows submissions using both the text box below and attached documents. Type your comments in the box below and use the Add Attachments button to include other documents. Save frequently while working.