METU Department of Computer Engineering Spring 2011

CENG 462—Artificial Intelligence

Homework 2-3 May 2, 2011

F.Polat

Implement a theorem prover for First Order Predicate Logic using *resolution refutation* technique and the *Set of Support* strategy. This program gets two lists of clauses; the list of base clauses and the list of clauses obtained from the negation of the theorem. Your program has to eliminate

- tautologies and
- subsumptions.

Your program returns whether the theorem is derivable, or not. If derivable, it has to print the resolutions that contribute to the proof of the theorem.

You will write a function called `prove` which will take the set of clauses and the negation of the theorem to be proved, and return whether the theorem is provable from these, and if so, the steps of the proof.

Input

Your `prove` function takes two parameters. The first parameter is the list of clauses in the base set and the second parameter is the list of clauses obtained from the negation of the theorem, respectively. Each clause will be given as a list. Variables, predicate names and function names start with a lower case letter, while constants start with and upper case letter. If the predicate is negated, the first element of the list is `~` and the second element is the name of the predicate, otherwise the first element is the name of the predicate.

The remaining items in the list are predicate arguments. An argument can be a constant, a variable or a function. If a predicate argument is a list, that means this parameter is a function. The first element of this list is the function name, and the remaining elements are the arguments to the function. See sample input and output section for clauses and their corresponding representations in your homework.

Output

Your `prove` function will return a list. If the theorem is provable, then the first element of the list should be `yes`, otherwise `no`. If the answer is yes, then the remaining items should contain the solution steps, where each solution step is a list consisting of three elements. The first two are the parent clauses and the third is the resolvent. If the answer is no. The list should not contain anything else.

Sample Input and Output

Clauses

$\{p(A, f(t))\}, \{q(z), \neg p(z, f(B))\}, \{\neg q(y), r(y)\}$

Negated theorem

$\{\neg r(A)\}$

Solution

$\{\neg r(A)\}, \{\neg q(y), r(y)\} \rightarrow \{\neg \mathbf{q}(\mathbf{A})\}$
$\{\neg q(A)\}, \{q(z), \neg p(z, f(B)\} \rightarrow \{\neg \mathbf{p}(\mathbf{A}, \mathbf{f}(\mathbf{B}))\}$
$\{\neg p(A, f(B))\}, \{p(A, f(T))\} \rightarrow \emptyset$

For the above example, your prove function will be called like this (indentation is provided for clarity):

```
> (prove '(((p A (f t))) ((q z) (~  p z (f B))) ((~ q y) (r y)))
         '(((~ r A))))
(yes
 (
  ((~ r A))
  ((~ q y) (r y))
  ((~ q A))
 )
 (
  ((~ q A))
  ((q z) (~ p z (f B)))
  (~ p A (f B))
 )
 (
  ((~ p A (f B)))
  ((p A (f T)))
  ()
 )
)
```

Submission Policy

- You are going to submit a single file called `hw2.cl` through COW System.
- Your `hw2.cl` should contain at least one function, i.e. `prove`.
- Your codes will be graded on inek machines. Make sure they run on these correctly.
- Late submissions will not be accepted.
- All work must be done individually and in compliance with course policy.