CSE 259 - Logic in Computer Science

Recitation-10

Project 3: Wang and Kobsa's Algorithm

Waqar Hassan Khan

Project 3

- The partial implementation **wangs_algorithm.pl** is runnable
- Sample input/output:
  - Premises as list ([p ^ q])
  - A formula as conclusion
  - We can derive the conclusion from the premise

```
?- run([p^q], [p]).
[p^q]=>[p]
=         [p,q]=>[p]          (by and/left)
=         Done (sharing a/f)

We can derive the conclusion from the premises.
true.
```

Project 3

- Write your codes here
- I have implemented some. The rest is your task

```
%%%%%%%%%%%%%%%% YOUR CODE STARTS %%%%%%%%%%%%%%%%

% Implement all other non-branching rules below by following Wangs algorithm

> /* ...
> prove(L => R):- ...
> prove(L => R):- ...
> /* ...
> prove(L => R) :- ...
> prove(L => R) :- ...
% Implement all branching rules below by following Wangs algorithm
> /* ...
> prove(L => R) :- ...
%%%%%%%%%%%%%%%% YOUR CODE ENDS %%%%%%%%%%%%%%%%
```

Project 3

- Two types of rules
  - Non-branching
  - Branching

Project 3: Non-branching Rules

Rule-1

If one of the formulae separated by commas is the negation of a formula, drop the negation sign and move it to the other side of the arrow.

Example:

Formula: p, ~(q ^ r) => p ^ r

Change to: p => p ^ r, q ^ r

Project 3: Non-branching Rules

Rule-2

If the last connective of a formula on the left is ^ (and), or on the right of the arrow is v (or), replace the connective by a comma.

Example-1:

Formula: p, p ^ q => r, s

Change to: p, p, q => r, s

Example-2:

Formula: p, q => r v p

Change to: p, q => r, p

Project 3: Non-branching Rules

Rule-3

If the last connective of a formula on the right is A → B, remove A → B from the right and then add A to the left and B to the right.

.

Example-1:

Formula: p v q => q → p

Change to: p v q, q => p

Project 3: Branching Rules

Rule-4

If the last connective of a formula on the left is v (or), or on the right of the arrow is ^ (and), then produce two new lines, each with one of the two sub formulae replacing the formula. Both of these must be proved in order to prove the original theorem.

Example-1:

Formula: p v q, r, s => q v p

Change to:

p, r, s => q v p

q, r, s => q v p

Example-2:

Formula: p, r, s => q ^ p

Change to:

p, r, s => q

p, r, s => p

Project 3: Branching Rules

Rule-5

If the last connective of a formula on the left is A → B, remove A → B from the left and then create two new lines, one with B added to the left, and the other with A added to the right.

Example-1:

Formula: p, p → q => p v q

Change to:

p, q => p v q

p => p v q, p

Project 3: Non-branching Rules – Rule 1

```
/*
 * Rule-1
 * example rule: negation
 * non-branching rule
 * If one of the formulae separated by commas is the
 * negation of a formula, drop the negation sign and
 * move it to the other side of the arrow.
 */
prove(L => R):-
    member(~X, L),
    del(~X, L, NewL),
    nl, write('=\t'), write(NewL => [X | R]),
    write('\t (by negation/left)'),
    prove(NewL => [X | R]).
prove(L => R):-
    member(~X, R),
    del(~X, R, NewR),
    nl, write('=\t'), write([X | L] => NewR),
    write('\t (by negation/right)'),
    prove([X | L] => NewR).
```

Example:

Formula: p, ~(q ^ r) => p ^ r

Change to: p => p ^ r, q ^ r

Output:

```
?- prove([p, ~(q ^ r)] => [p ^ r]).
=        [p]=>[q^r,p^r]    (by negation/left)
```

Project 3: Non-branching Rules – Rule 2

```
/*
 * Rule-2
 * example rule: left conjuction
 * non-branching rule
 * If the last connective of a formula on the left is ^ (and),
 * or on the right of the arrow is v (or), replace the connective by a comma.
 */
prove(L => R) :-
  member(A ^ B, L),
  del(A ^ B, L, NewL),
  nl, write('=\t'), write([A, B | NewL] => R),
  write('\t (by and/left)'),
  prove([A, B | NewL] => R).
prove(L => R) :-
  member(A v B, R),
  del(A v B, R, NewR),
  nl, write('=\t'), write(L => [A, B | NewR]),
  write('\t (by or/right)'),
  prove(L => [A, B | NewR]).
```

Project 3: Non-branching Rules – Rule 2 contd.

Example-1:

Formula: p, p ^ q => r, s

Change to: p, p, q => r, s

```
?- prove([p, p ^ q] => [r, s]).

=         [p,q,p]=>[r,s]    (by and/left)
```

Example-2:

Formula: p, q => r v p

Change to: p, q => r, p

```
?- prove([p, q] => [r v p]).

=         [p,q]=>[r,p]       (by or/right)
=         Done (sharing a/f)
```

Project 3: Branching Rules – Rule 5

```
/*
 * Rule-5
 * example rule: left implication
 * branching rule
 * If the last connective of a formula on the left is A → B,
 * remove A → B from the left and then create two new lines,
 * one with B added to the left, and the other with A added to the right.
 */
prove(L => R) :-
    member(A -> B, L),
    del(A -> B, L, NewL),
    nl,
    write('\tFirst branch: '),
    nl,
    write('=\t'),
    write([B | NewL] => R),
    write('\t (by arrow/left)'),
    prove([B | NewL] => R),
    nl,        You, 1 second ago • Uncommitted changes
    write('\tSecond branch: '),
    nl,
    write('=\t'),
    write(NewL => [A | R]),
    write('\t (by arrow/left)'),
    prove(NewL => [A | R]).
```

Project 3: Your Tasks

- Rule-3: Non-branching rule
- Rule-4: Branching rule