02156 Exercises-06

Jørgen Villadsen

2021

Exercise 1

Consider the following formula: $(p \to q \to r) \to (\neg q \to r) \to (\neg p \to r) \to r$

Use refutation and the resolution procedure (if you are uncertain about this then start with the formulas $p \to q \to p$ and $(p \to q) \to p$ instead). Try two refutations: one by resolving the literals in the order p, q, r and the other in the order r, q, p.

State whether this shows that the formula is valid or not.

When you have *manually* finished the two refutations you can use the program `resolution(+XFml)` that succeeds iff `XFml` can be refuted (`XFml` is a formula in external format).

```
?- resolution( (p => q => r) => (~q => r) => (~p => r) => r ).
```

The program is available in the file `resolution.pl` on CampusNet (Program files folder in the top folder) and prints the results in clausal form. Which order of the literals is used by the program `resolution` in this case?

Study everything in the file `resolution.pl` in details — at home — and test the following formulas:

$p \to q \to p$ $\quad (p \to q) \to p$ $\quad p$ $\quad p \leftrightarrow p$ $\quad p \leftrightarrow p \leftrightarrow p$ $\quad p \leftrightarrow p \leftrightarrow p \leftrightarrow p$ $\quad p \leftrightarrow p \leftrightarrow p \leftrightarrow p \leftrightarrow p$ $\ldots$

Exercise 2

Consider the following program `add(+Elem,+List1,?List2)` that succeeds iff adding `Elem` once to `List1` gives `List2` (assuming that it is not to be added if already there).

```
add(X,Y,Z) :- member(X,Y), !, Z = Y.
add(X,Y,[X|Y]).
```

Imagine that the program is changed to the following incorrect program:

```
add(X,Y,Y) :- member(X,Y), !.
add(X,Y,[X|Y]).
```

Find a simple query that shows that it is incorrect.

Exercise 3

Use the if-the-else construction in a program `minmax` that takes two numbers and returns both the minimum and maximum of these numbers as follows:

```
?- minmax(1,2,X,Y).

X = 1
Y = 2

Yes

?- minmax(2,1,X,Y).

X = 1
Y = 2

Yes
```

Exercise 4

Consider the following program `tautology(+XFml)` that succeeds iff `XFml` is a tautology (`XFml` is a formula in external format).

Rewrite the program such that it uses the negation-as-failure operator `\+` instead of the cut (`!`).

```
:- ensure_loaded(logic).

tautology(XFml) :-
  to_internal(XFml,Fml), get_atoms(Fml,Atoms),
  generate(Atoms,V), tt(Fml,V,f), !, fail.
tautology(_).
```

Exercise 5

A team is a set of students represented as a list of student numbers.

Write a program `test(+Team1,+Team2,?Team3)` that unifies `Team3` with `Team1` if `Team1` and `Team2` does not have a student in common and with `[]` otherwise.

Exercise 6

Write a program `ms(+List1,?List2)` that succeeds if and only if `List2` is a permutation of `List1` and the elements in `List2` are ordered by the `=<` relation.

Use the Mergesort algorithm, that is, split in non-empty halves, sort them, and then do a merge.

Exercise 7

Consider the following Prolog program code:

```
p(X) :- q(X,f(X)).

q(a,f(a)).
q(_,Y) :- r(g(Y)).
q(_,_).
q(Z,Z) :- !, fail.
q(e,f(e)).

r(g(f(b))).
r(f(g(c))).
r(g(d)).
```

State the result of the following query without using a computer:

```
?- p(X).
```

Exercise 8

Consider the following formula:

$$\neg(p \ \wedge \ (p \rightarrow ((q \vee r) \ \wedge \ \neg(q \wedge r))) \ \wedge \ (p \rightarrow ((s \vee t) \wedge \neg(s \wedge t))) \\ \wedge \ (s \rightarrow q) \ \wedge \ (\neg r \rightarrow t) \ \wedge \ (t \rightarrow s))$$

Use refutation and the resolution procedure.

State whether this shows that the formula is valid or not.

Hint: Use the following equivalences:

$$\begin{aligned} A \rightarrow ((B \vee C) \wedge \neg(B \wedge C)) \ &\equiv \ \neg A \vee ((B \vee C) \wedge (\neg B \vee \neg C)) \\ &\equiv \ (\neg A \vee B \vee C) \wedge (\neg A \vee \neg B \vee \neg C) \end{aligned}$$