
Logic in Computer Science, October 31st, 2023. Time: 1h45min. No books or lecture notes allowed.

```
 - Insert your answers on the dotted lines ... below, and only there.
 - When finished, upload this file with the same name: exam.txt
 - Use the text symbols:     &     v    -      ->          |=        A       E
     for                    AND   OR   NOT   IMPLIES   "SATISFIES"  FORALL  EXISTS  etc.,  like in:
    I |=  p & (q v -r)     (the interpretation I satisfies the formula p & (q v -r)  ).
   You can write  not (I |= F)  to express "I does not satisfy F",  or
                  not (F |= G)  to express "G is not a logical consequence of F"
   Also you can use subindices with "_". For example write x_i to denote x-sub-i.
```

Problem 1. (2.5 points).

1a) Is it true that if F -> G is a tautology then G |= F? Prove using only the definition of propositional logic.

>>> Answer:
...

1b) Is it true that for any two propositional formulas F and G, we have that -F v G is a tautology if and only if F |= G? Prove it using only the definition of propositional logic.

>>> Answer:
...

Problem 2. (2.5 points).

In what follows, you can consider proved, and you can use it in a prove, the following statement:

(ST) Let F1,F2,F1',F2' be formulas. If F1 |= F1' and F2 |= F2', then
F1 & F2 |= F1' & F2' and F1 v F2 |= F1' v F2'.

2a) Let F be a formula where v (or) and & (and) are the only connectives. Show that if in F we replace an occurrence of a subformula G with another G' such that G |= G', we obtain a new subformula F' such that F |= F' (F' is a logical consequence of F) Prove the statement by *induction* on n(F), defined as the number of connectives of F minus the number of connectives of G (the subformula).

>>> Answer
* Base case: If n(F) = 0,
...

* Inductive case: suppose that n(F) > 0, and that the statement is true for any formula H such that H contains an occurrence of G as a subformula, and n(H) < n(F).
...

2b) Can the previous result be extended to formulas F where the connective - (not) can also appear? Justify the answer.

>>> Answer:
...

Problem 3. (2.5 points).

3a) What is Horn-SAT? What is its computational complexity? Explain very briefly why.

>>> Answer:
...

3b) Let S be a set of propositional clauses over a set of n predicate symbols, and let Res(S) be its closure under resolution. For each one of the following cases indicate whether Res(S) is infinite or finite, and, if finite, of which size. Consider that each clause is a set (i.e., no repetitions) of literals.

* If clauses in S have at most two literals.

>>> Answer:
...

* Every clause in S has either two literals or is a Horn clause.

>>> Answer:
...

Problem 4. (2.5 points).

4) Consider the following C++ code snippet:

```
1: int f(int i, int s, const vector<int>& v) {
2:    if (s == 0) return 1;
3:    while (i < 0 and s != 0) ++i;
4:    if (i >= v.size()) return -1;
5:    return v[i];
6: }
```

Let us introduce propositional symbols p, q, r with the following meaning:

```
p:  "i >= 0       at line 5"
q:  "i < v.size() at line 5"
r:  "s == 0"
```

We note that the value of program variable s never changes in the code,
so in the definition of propositional symbol r we do not need to indicate
which line we are referring to.

4a) Give a propositional formula in CNF such that if it is unsatisfiable then we can ensure that
the vector access v[i] at line 5 is correct.

>>> Answer:
At line 5 we have -r, because if s == 0 we would have returned at line 2.
We also have
...

On the other hand, the vector acces v[i] at line 5 is correct if and only if
...

So if we prove that
... |= ...
then the vector access will be correct.

But this is equivalent to proving that
...
is unsatisfiable.

4b) Prove that indeed the access v[i] at line 5 is correct.

>>> Answer:
...