Lógica en la Informática / Logic in Computer Science

Tuesday January 13th, 2015

Time: 2h30min. No books, lecture notes or formula sheets allowed.
The propositional logic part comprises questions 1 to 4.

1) Let F be a propositional formula. Is it true that F is a tautology if and only if $\neg F$ is unsatisfiable? Prove it using only the definition of propositional logic.

Solution: Yes. F tautology iff, by definition, $\forall I,\ eval_I(F) = 1$ iff $\forall I,\ 1 - eval_I(F) = 0$, iff, by definition of evaluation of $\neg F$, $\forall I,\ eval_I(\neg F) = 0$ iff $\forall I,\ I \not\models \neg F$ iff $\neg F$ unsatisfiable.

2) How would you efficiently decide whether a given propositional formula in DNF is satisfiable?

Solution: A formula in DNF is a disjunction of *cubes* c, where each cube is a conjunction of literals. Such a DNF $c_1 \vee \ldots c_n$ is satisfiable iff exists interpretation I s.t. $I \models c_1 \vee \ldots \vee c_n$ iff exists I $eval_I(c_1 \vee \ldots \vee c_n) = 1$ iff exists I $max(eval_I(c_1), \ldots eval_I(c_n)) = 1$, that is, if at least one of the cubes c_i is satisfiable. This can be checked as follows: a cube $l_1 \wedge \ldots \wedge l_m$ is satisfiable if for some I all lits literals are true in I. It is easy to see that such an I exists iff $\{l_1 \ldots l_m\}$ contains no pair of complementary literals, of the form p and $\neg p$ (if there is no such a pair then I can simply set the positive literals of the cube to 1 and the negative ones to 0). This can be checked in linear time.

3) We want to solve a certain problem with a SAT solver. It contains a number of at-most-one constraints of the form $x_1 + \ldots + x_n \leq 1$, each one of which has to be encoded (expressed as a set of clauses). A good property of such an encoding is that, as soon as one of the variables x_i becomes true, the unit propagation mechanism of the SAT solver will set to false all the other variables $x_1 \ldots x_{i-1}, x_{i+1} \ldots x_n$. Does the Heule-3 encoding have this property? Briefly explain why.

Solution: The *Heule-3 encoding* uses the fact that $amo(x_1 \ldots x_n)$ iff $amo(x_1, x_2, x_3, aux)$ AND $amo(\neg aux, x_4 \ldots x_n)$. Then the part $amo(\neg aux, x_4 \ldots x_n)$, which has $n-2$ variables, can be encoded recursively in the same way, and $amo(x_1, x_2, x_3, aux)$ can be expressed using the quadratic encoding with 6 clauses. In this way, for eliminating two variables we need one auxiliary variable end six clauses, so in total we need $n/2$ variables and $3n$ clauses.

The Heule-3 encoding does have the property. We can prove it, for example, by induction on n.

Base case: if $n \leq 4$ the quadratic encoding part is the whole constraint. For example, for $n = 4$ we have $\neg x_1 \vee \neg x_2,\ \neg x_1 \vee \neg x_3,\ \neg x_1 \vee \neg x_4,\ \neg x_2 \vee \neg x_3,\ \neg x_2 \vee \neg x_4$, and $\neg x_3 \vee \neg x_4$. For every distinct pair $i, j \subset \{1 \ldots 4\}$ we have a clause $\neg x_i \vee \neg x_j$, so if x_i becomes true, all other variables x_j become false by unit propagation.

Induction case: If $n > 4$ the part $amo(x_1, x_2, x_3, aux)$ is expressed using the quadratic encoding with 6 clauses: $\neg x_1 \vee \neg x_2,\ \neg x_1 \vee \neg x_3,\ \neg x_1 \vee \neg aux,\ \neg x_2 \vee \neg x_3,\ \neg x_2 \vee \neg aux$, and $\neg x_3 \vee \neg aux$. Now there are two cases: A) some variable of $\{x_1, x_2, x_3\}$ becomes true, or B) some variable of $x_4 \ldots x_n$ becomes true. In case A, as before, unit propagation will set the other variables in $\{x_1, x_2, x_3, aux\}$ to false, and hence $\neg aux$ becomes true, and then we can apply the induction hypothesis since $amo(\neg aux, x_4 \ldots x_n)$ has two variables less: unit propagation will set all variables in $\{x_4 \ldots x_n\}$ to false.

In case B, by induction hypothesis, since $amo(\neg aux, x_4 \ldots x_n)$ has two variables less, unit propagation will set all other variables in $\{x_4 \ldots x_n, \neg aux\}$ to false and hence aux becomes true, and by the clauses $\neg x_1 \vee \neg aux,\ \neg x_2 \vee \neg aux,\ \neg x_3 \vee \neg aux,$ unit propagation will set x_1, x_2, and x_3 to false.

4) The Seat car manufacturer has a website where one can order a car, *configuring* all its (hundreds of) *properties*: which type of engine you want, which color, type of music equipment, etc. The configuration software has to take into account that there exist constraints relating pairs of properties, e.g., "this type of air conditioning cannot go together with that kind of diesel engine". In general such constraints have the form: "property P_i cannot go together with property P_j", or "if property P_i is true then also property P_j must be true", or "at least one of property P_i and property P_j must be true". The software

should give a warning if the user tries to make a configuration that is forbidden with respect to these constraints.
4a) Explain how you would do that using propositional logic (which variables and clauses you would introduce, etc.).

Solution: Assume there are N different properties. For each property $i \in 1 \ldots 100$ we introduce the variable x_i, meaning "property i is included". Now for each constraint relating two properties i and j, it suffices to have a binary clause of one of the following four forms: $x_i \vee x_j$, or $x_i \vee \neg x_j$, or $\neg x_i \vee x_j$, or $\neg x_i \vee \neg x_j$. Each property i the user selects will be a 1-literal clause x_i. The whole problem of checking if a configuration is allowed or forbidden is hence polynomial: a 2-SAT problem, that can be decided in linear time (resolution solves it in quadratic time).

4b) How would you handle constraints relating more than two properties, such as "the user must choose at least one of the three properties P_i, P_j or P_k"?

Solution: This requires having 3-literal clauses, and the problem becomes NP-hard, so in principle one cannot do better than using a general-purpose SAT solver.

5) Consider two groups of 10 people each. In the first group, as expected, the percentage of people with lung cancer among smokers is higher than among non-smokers. In the second group, the same is the case. But if we consider the 20 people of the two groups together, then the situation is the opposite: the proportion of people with lung cancer is higher among non-smokers than among smokers! Can this be true? Write a little Prolog program to find it out.
Solution:

```
num(X):- between(1,7,X).  % below, e.g. SNC1 denotes "num. smokers with no cancer group 1".

p:- num(SC1), num(SNC1),  num(NSC1), num(NSNC1), 10 is SC1+SNC1+NSC1+NSNC1,
    SC1/(SC1+SNC1) > NSC1/(NSC1+NSNC1),
    num(SC2), num(SNC2),  num(NSC2), num(NSNC2), 10 is SC2+SNC2+NSC2+NSNC2,
    SC2/(SC2+SNC2) > NSC2/(NSC2+NSNC2),
    (SC1+SC2)/(SC1+SNC1+SC2+SNC2) < (NSC1+NSC2)/(NSC1+NSNC1+NSC2+NSNC2),
    write([ SC1,SNC1,NSC1,NSNC1,SC2,SNC2,NSC2,NSNC2]), nl, halt.

%                 smokers            non-smokers
%            cancer  non-cancer   cancer  non-cancer
%  ------------------------------------------------
%  group 1:    2         1          4        3          2/3 > 4/7
%  group 2:    2         4          1        3          2/6 > 1/4
%  total       4         5          5        6          4/9 < 5/11

% another solution allowing zeroes:
% group1:      1         0          2        1          1/1 > 2/3
% group2:      1         3          0        1          1/4 > 0/1
% total:       2         3          2        2          2/5 < 2/4
```

6) Let F be the first-order formula $\exists x \forall y \exists z\ p(z,y) \wedge \neg p(x,y)$.
6a) Give a model I with $D_I = \{a, b, c\}$.
Solution: Intuitively, we can build the model considering. e.g., that the x that exists is a. Then we need that $\neg p(x,y)$ for all y, that is, $p_I(a,a) = 0,\ p_I(a,b) = 0,\ p_I(a,c) = 0$. Furthermore, we need that for all y, there exists a z such that $p(z,y)$, which we can achieve by taking always the same z (this is not necessary, but here it works): $p_I(b,a) = 1,\ p_I(b,b) = 1,\ p_I(b,c) = 1$. This gives us a model independently of how we define the remaining three cases $p_I(c,a),\ p_I(c,b),\ p_I(c,c)$.

6b) Is it true that $F \models \forall x\ p(x,x)$?
Solution: No. The model of F given in 6a does not satisfy $\forall x\ p(x,x)$.

6c) Is there any model of F with a single-element domain?
Solution: No. Calling that single element a, i.e., $D_i = \{a\}$, we would need $p_I(a,a) = 1$ due to the subformula $p(z,y)$, but also $p_I(a,a) = 0$ due to the subformula $\neg p(x,y)$.

7) In a certain village, the barber shaves all men that do not shave themselves, and only these men. Formalize this sentence in first-order logic and prove by resolution that the barber is a woman (i.e., not a man).

Solution: Let the constant b^0 represent the barber, let $sh(x, y)$ mean "x shaves y", and let $m(x)$ mean "x is a man". Then the sentence F becomes $\forall x\ (sh(b,x)\ \leftrightarrow\ (m(x) \wedge \neg sh(x,x)))$. We need to prove that $F \models \neg m(b)$, that is, $F \wedge m(b)$ is unsatisfiable. Transforming F into clausal form:

$\forall x\ (\neg sh(b,x)\ \vee\ (m(x) \wedge \neg sh(x,x)))\ \ \wedge\ \ (sh(b,x)\ \vee\ \neg(m(x) \wedge \neg sh(x,x)))$

$\forall x\ (\neg sh(b,x)\ \vee\ (m(x) \wedge \neg sh(x,x)))\ \ \wedge\ \ (sh(b,x)\ \vee\ \neg m(x)\ \vee\ sh(x,x))$

$\forall x\ (\neg sh(b,x)\ \vee\ m(x))\ \ \wedge\ \ (\neg sh(b,x)\ \vee\ \neg sh(x,x))\ \ \wedge\ \ (sh(b,x)\ \vee\ \neg m(x)\ \vee\ sh(x,x))$

And we get the clauses 1-4, which give the empty clause by resolution as follows:

1. $m(b)$
2. $\neg sh(b,x)\ \vee\ m(x)$
3. $\neg sh(b,x)\ \vee\ \neg sh(x,x)$
4. $sh(b,x)\ \vee\ \neg m(x)\ \vee\ sh(x,x)$
5. $sh(b,b)$ $\qquad 1+4 \quad \{x=b\}$
6. $\neg sh(b,b)$ $\qquad 3+5 \quad \{x=b\}$
7. *emptyclause* $\qquad 5+6 \quad \{\ \}$