Last names: ... **1st name: ...** **DNI: ...**

Lgica en la Informtica / Logic in Computer Science

Permutation B. Tuesday April 18th, 2017
Time: 1h45min. No books, lecture notes or formula sheets allowed.

1) Let us remember the *Heule-3 encoding* for *at-most-one* (*amo*) that is, for expressing in CNF that at most one of the literals $x_1 \dots x_n$ is true, also written $x_1 + \dots + x_n \leq 1$. It uses the fact that $amo(x_1 \dots x_n)$ iff $amo(x_1, x_2, x_3, aux)$ AND $amo(\neg aux, x_4 \dots x_n)$. Then the part $amo(\neg aux, x_4 \dots x_n)$, 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.

1a We now want to extend the encoding for *at-most-two* (*amt*, also written $x_1 + \dots + x_n \leq 2$). Prove that $amt(x_1 \dots x_n)$ has a model I iff $amt(x_1, x_2, x_3, aux_1, aux_2) \wedge amt(\neg aux_1, \neg aux_2, x_4 \dots x_n)$ has a model I', with $I(x_i) = I'(x_i)$ for all i in $1 \dots n$.

Answer:
$\Longrightarrow$: If $I \models amt(x_1 \dots x_n)$ and k is the number of literals of $\{x_1, x_2, x_3\}$ that are true in I, then we extend I into I' as follows: if $k = 0$ we set $I'(aux_1) = I'(aux_2) = 1$; if $k = 1$ we set (for example) $I'(aux_1) = 1$ and $I'(aux_2) = 0$; if $k = 2$ we set $I'(aux_1) = I'(aux_2) = 0$. In all three cases $I' \models amt(x_1, x_2, x_3, aux_1, aux_2) \wedge amt(\neg aux_1, \neg aux_2, x_4 \dots x_n)$.
$\Longleftarrow$: If $I' \models amt(x_1, x_2, x_3, aux_1, aux_2) \wedge amt(\neg aux_1, \neg aux_2, x_4 \dots x_n)$ then, "forgetting" the part of the auxiliary variables, in all cases the resulting I is a model of $amt(x_1 \dots x_n)$, because:

- if $I'(aux_1) = I'(aux_2) = 1$ then $I \models \neg x_1 \wedge \neg x_2 \wedge \neg x_3$ and $I \models amt(x_4 \dots x_n)$
- if $I'(aux_1) = I'(aux_2) = 0$ then $I \models amt(x_1, x_2, x_3)$ and $I \models \neg x_4 \wedge \dots \wedge \neg x_n$
- if $I'(aux_1) = 0$ and $I'(aux_2) = 1$ (or vice versa) then $I \models amo(x_1, x_2, x_3)$ and $I \models amo(x_4 \dots x_n)$.

1b Write all clauses for encoding $amt(x_1, x_2, x_3, aux_1, aux_2)$ with no more auxiliary variables.

Answer: We need one clause for each subset of 3 elements out of 5, that is, $\binom{5}{3} = 10$ clauses:
$\neg x_1 \vee \neg x_2 \vee \neg x_3$, $\neg x_1 \vee \neg x_2 \vee \neg aux_1$, $\neg x_1 \vee \neg x_2 \vee \neg aux_2$, $\neg x_1 \vee \neg x_3 \vee \neg aux_1$, $\neg x_1 \vee \neg x_3 \vee \neg aux_2$, $\neg x_1 \vee \neg aux_1 \vee \neg aux_2$, $\neg x_2 \vee \neg x_3 \vee \neg aux_1$, $\neg x_2 \vee \neg x_3 \vee \neg aux_2$, $\neg x_2 \vee \neg aux_1 \vee \neg aux_2$, $\neg x_3 \vee \neg aux_1 \vee \neg aux_2$.

1c How many clauses and auxiliary variables are needed in total for $amt(x_1 \dots x_n)$ in this way?

Answer: The part $amt(\neg aux_1, \neg aux_2, x_4 \dots x_n)$ has one literal less. So to eliminate one literal, we need 10 clauses and 2 auxiliary variables and hence in total $10n$ clauses and $2n$ auxiliary variables.

1d The Heule-3 encoding for $amo(x_1, \dots, x_n)$ has a good property: if one of the literals x_i becomes true, all other literals in $x_1, \dots, x_n$ are set to false **by unit propagation**. Does this *amt* encoding have such a property?, that is, if two of $x_1 \dots x_n$ become true, will unit propagation set the other variables to false? Explain why.

Answer: No. For example, if x_1 and x_4 become true, no unit propagation takes place at all.

2) Every propositional formula F over n variables can also expressed by a Boolean circuit with n inputs and one output. In fact, sometimes the circuit can be much smaller than F because each subformula only needs to be represented once. For example, if F is

$$x_1 \wedge (x_3 \wedge x_4 \ \vee \ x_3 \wedge x_4) \quad \vee \quad x_2 \wedge (x_3 \wedge x_4 \ \vee \ x_3 \wedge x_4),$$

a circuit for F with only five gates, representing the output of each logical gate as a new variable (a natural number, and using 0 as the output), is:

```
0 = or(1,2)        1 = and(x1,3)        3 = or(4,4)
                   2 = and(x2,3)        4 = and(x3,x4)
```

Explain **very briefly** how you would use a standard SAT solver for CNFs to **efficiently** determine whether two circuits C_1 and C_2, represented like this, are logically equivalent.

Answer: We can apply the Tseitin transformation directly to each sub-circuit: each gate already has its auxiliary variable. Each gate $n = and(x, y)$, generates three clauses: $\neg n \vee x$, $\neg n \vee y$, and $n \vee \neg x \vee \neg y$, and each gate $n = or(x, y)$ another three: $n \vee \neg x$, $n \vee \neg y$, and $\neg n \vee x \vee y$. Negations can also be handled as usual. Let S_1 and S_2 be the resulting sets of clauses for the gates of C_1 and C_2, respectively, using different names $0', 1', 2' \ldots$ for the auxiliary variables of C_2. Then we have:

$C_1 \equiv C_2$ (both circuits have the same models) iff
there is no model of $S_1 \cup S_2$ such that the root variables 0 and $0'$ get different values iff
on (CNF) input $S_1 \cup S_2 \cup \{ \neg 0 \vee \neg 0', \ \ 0 \vee 0' \}$, the SAT solver returns unsatisfiable.

Note: if we first transform the circuits (directed acyclic graphs) into formulas (trees) and then apply Tseitin, the CNF can become much larger, due to multiple copies of sub-circuits.

3) For each one of the following statements, indicate here whether it is true or false without giving any explanations why.

1. If F is unsatisfiable, then for every G we have $G \models F$. **False**
2. If F is unsatisfiable, then for every G we have $F \models G$. **True**
3. Let F, G, H be formulas. If $F \vee G \models H$ then $F \wedge \neg H$ is unsatisfiable. **True**
4. The formula $p \vee p$ is a logical consequence of the formula $(p \vee q \vee r) \wedge (\neg q \vee r) \wedge (\neg r)$. **True**
5. The formula $(p \vee q) \wedge (\neg p \vee q) \wedge (\neg p \vee \neg q) \wedge (\neg q \vee p)$ is unsatisfiable. **True**
6. If F is a tautology, then for every G we have $F \models G$. **False**
7. Let F, G, H be formulas. If $F \wedge G \not\models H$ then $F \wedge G \wedge H$ is unsatisfiable. **False**
8. Let F, G, H be formulas. If $F \wedge G \models \neg H$ then $F \wedge G \wedge H$ is unsatisfiable. **True**
9. If F es a tautology, then for every G we have $G \models F$. **True**
10. Assume $|\mathcal{P}| = n$. There are 2^n interpretations. Moreover there are exactly $k = 2^{2^n}$ formulas $F_1, \ldots, F_k$ such that for all i, j with $i \neq j$ in $1 \ldots k$, $F_i \not\equiv F_j$. Each one of these F_i represents a different Boolean function. **True**