Lógica en la Informática / Logic in Computer Science

Tuesday April 21st, 2015

Time: 1h45min. No books, lecture notes or formula sheets allowed.

1) Which is the minimal known computational cost of deciding the satisfiability of the following types of propositional formulas (along this whole exam: just write the answer; do not explain why, unless you are asked to do so explicitly):

a: Arbitrary propositional formulas: `Answer: NP-complete (so, best known=exponential)`

b: 3-SAT: `Answer: NP-complete`

c: 2-SAT: `Answer: polynomial (linear)`

d: DNF; `Answer: polynomial (linear)`

e: CNF; `Answer: NP-complete`

f: Horn-SAT: `Answer: polynomial (linear)`

2)

2a: What is the minimal known computational cost of deciding whether a given propositional formula F is a tautology? `Answer: NP-complete`

2b: What is the minimal known computational cost of deciding whether a given propositional formula F in CNF is a tautology? Why?

`Answer:` Linear: F is a tautology iff each clause C in F is a tautology. And a clause is a tautology iff it contains two literals of the form p and $\neg p$, for some p, which is easy to check in linear time.

2c: What is the minimal known computational cost of transforming a given propositional formula F into a logically equivalent formula G in CNF, without using any new auxiliary variables?

`Answer:` Exponential.

2d: Write the result of applying the Tseitin transformation to $p \vee (q \wedge r) \vee (\neg p \wedge r)$.

`Answer:` Let us consider that the leftmost $\vee$ is the topmost connective. Introduce a new symbol a_1 for this $\vee$, a_2 for the second $\vee$, and a_3, a_4 for the two $\wedge$s. The clauses become: a_1 (the root symbol),

three clauses for $a_1 \equiv p \vee a_2$: $\neg p \vee a_1,\ \neg a_2 \vee a_1,\ \neg a_1 \vee p \vee a_2$.
three clauses for $a_2 \equiv a_3 \vee a_4$: $\neg a_3 \vee a_2,\ \neg a_4 \vee a_2,\ \neg a_2 \vee a_3 \vee a_4$.
three clauses for $a_3 \equiv q \wedge r$: $\neg a_3 \vee q,\ \neg a_3 \vee r,\ \neg q \vee \neg r \vee a_3$.
three clauses for $a_4 \equiv \neg p \wedge r$: $\neg a_4 \vee \neg p,\ \neg a_4 \vee r,\ p \vee \neg r \vee a_4$.

2e: Let G be the Tseitin transformation of a given propositional formula F. What is the size of G with respect to F? What are the properties of G with respect to F?

`Answer:` The size of G is linear w.r.t. F: three clauses for each binary connective in F. G is equisatisfiable to F, that is, G is satisfiable iff F is satisfiable.

Remark: Note that there can be no Tseitin-like transformation converting F into a CNF G that is linear in the size of F and such that $G \equiv F$ or such that at least the property of being a tautology is preserved (G tautology iff F tautology): then the answer to **2a** would be "linear".

3) Let F and G be propositional formulas. For each of the following two statements, say whether it is right or wrong, and prove why:

3a: $F \models G$ implies $F \equiv F \wedge G$

Answer: This is true.
for all I, if $I \models F$ then $I \models G$, which means that
for all I, either $I \not\models F$, or $I \models F$ and $I \models G$, which implies (by def. of $\models$).

for all I, either $eval_I(F) = 0$ or $eval_I(F) = eval_I(G) = 1$. that is,
for all I, either $eval_I(F) = min(eval_I(F), eval_I(G)) = 0$ or $eval_I(F) = min(eval_I(F), eval_I(G)) = 1$, which implies (by def. of $eval_I$),
for all I, either $I \not\models (F)$ and $I \not\models F \wedge G$ or $I \models (F)$ and $I \models F \wedge G$,
that is, (by definition of logical equivalence) $F \equiv F \wedge G$.

3b: Always $F \models G$ or $F \models \neg G$.
Answer: This is false. Counterexample: Assume $F = p$ and $G = q$ for two distinct symbols p and q. Then $F \not\models G$, since for the interpretation I where $I(p) = 1$ and $I(q) = 0$, we have $I \models F$ but $I \not\models G$. And also $F \not\models \neg G$, since for the interpretation I' where $I'(p) = 1$ and $I'(q) = 1$, we have $I' \models F$ but $I' \not\models \neg G$.

4) A factory produces rolls of 100m of cable. The factory has n orders from customers. Each order i in $1..n$ is for buying k_i meters of cable, $k_i \leq 100$. All orders have to be served by cutting the pieces of 100m, without using more than K rolls.

4a: Explain in detail how to use a SAT solver for solving this. Clearly indicate which variables you use and their precise meaning, and which properties you impose using which clauses or which constraints (for cardinality or pseudo-Boolean constraints, it is not necessary to write their encodings into clauses).
Answer:

We introduce variables x_{ij} meaning "order i is cut from roll j" for i in $1..n$ and j in $1..K$.

Constraints/clauses: Each order i is cut from at least one roll: one clause $x_{i1} \vee \ldots \vee x_{iK}$ for each i in $1..n$.

Note that here it is not really necessary to impose that each order i is cut from at most one roll: if some order is served more than once, we can simply ignore all but one of them.

From no roll more than 100m cable can be cut: for each roll j in $1..K$, one pseudo-Boolean constraint: $k_1 * x_{1j} + \ldots + k_n * x_{nj} \leq 100$.

Remark: this is a well-known NP-complete problem (called the *cutting stock* problem).

4b: Answer the same questions, assuming that for each order i in $1..n$ we also know the price p_i (in euros) each customer will pay for its k_i meters of cable, and the factory wants to know whether it can obtain an income of at least 10.000 euros with its K rolls by serving *some of* the orders.
Answer:

We have the same variables as before.

Now we do not want to impose that each order is cut from at least one roll. But here we need to impose that each order i is cut from *at most one* roll, since otherwise we could get a solution adding up to 10.000 euros by serving the same order many times! For example, if some order needs only 10m but pays 1000 euros, we could simply serve that order ten times, which would be wrong. So for each order i in $1..N$, we need we need one constraint $AMO(x_{i1}, \ldots, x_{iK})$, that is, $x_{i1} + \ldots + x_{iK} \leq 1$.

From no roll more than 100m cable can be cut: for each roll j in $1..K$, one pseudo-Boolean constraint: $k_1 * x_{1j} + \ldots + k_n * x_{nj} \leq 100$.

Total income of at least 10.000 euros: one pseudo-Boolean constraint:

$p_1 * x_{11} + \ldots + p_1 x_{1K} \;+\; \ldots \;+\; p_n * x_{n1} + \ldots + p_n x_{nK} \;\geq\; 10000.$