Lógica en la Informática / Logic in Computer Science

Tuesday November 26th, 2013

Time: 1h55min. No books, lecture notes or formula sheets allowed.

1A) Given two propositional formulas F and G, is it true that $F \to G$ is a tautology iff $F \wedge \neg G$ is unsatisfiable? Prove it using only the formal definitions of propositional logic.

Solution: It is true:

$F \to G$ tautology		iff	[def. $\to$]
$\neg F \vee G$ tautology		iff	[def. tautology]
for every I,	$eval_I(\neg F \vee G) = 1$	iff	[def. $eval_I$ $\vee$]
for every I,	$max(eval_I(\neg F), eval_I(G)) = 1$	iff	[def. $eval_i$ $\neg$]
for every I,	$max(1 - eval_I(F), eval_I(G)) = 1$	iff	[arithmetic: $eval_I(F) = 0$ or $eval_I(G) = 1$]
for every I,	$min(eval_I(F), 1 - eval_I(G)) = 0$	iff	[def. $eval_I$ $\neg$]
for every I,	$min(eval_I(F), eval_I(\neg G)) = 0$	iff	[def. $eval_I$ $\wedge$]
for every I,	$eval_I(F \wedge \neg G) = 0$	iff	[def. unsat.]
$F \wedge \neg G$ unsatisfiable			

1B) Given two propositional formulas F and G, is it true that $F \models G$ iff $F \to G$ is satisfiable? Prove it using only the formal definitions of propositional logic.

Solution: It is false. A counterexample is as follows: let F be the formula p and G be the formula $\neg p$. The resulting formula $p \to \neg p$ is satisfiable ($I(p) = 0$ is a model), but $p \not\models \neg p$, as the interpretation $I(p) = 1$ is a model of p but not a model of $\neg p$.

1C) What is the complexity of Horn-SAT? What is the complexity of 2-SAT? What do you think is the complexity of HornOrTwo-SAT, that is, deciding the satisfiability of sets S of clauses such that every clause in S is either Horn or has at most two literals?

Solution: Horn-SAT and 2-SAT are both polynomial (linear, in fact). But if clauses of both kinds are allowed, then it becomes NP-Complete, because we can easily transform *any* clause set S into an equisatisfiable HornOrTwo-SAT clause set, as follows. For each variable x_i, we introduce a new variable x'_i and add two-literal clauses expressing that x'_i is the negation of x_i (this is similar to the Tseitin transformation): $\neg x_i \vee \neg x'_i$ and $x_i \vee x'_i$. After that we can convert all clauses of S into clauses with only negative literals (which are Horn clauses) by replacing, for each variable x_i, all positive literals x_i by $\neg x'_i$.

2) Consider the *at most one* constraint (AMO), saying that among n literals $l_1 \dots l_n$, at most one can be true, that is, $l_1 + \cdots + l_n \leq 1$.

2A) Write the name of the encoding you know for this constraint that needs the fewest auxiliary variables. How many (as a function of n)?

Solution: The quadratic encoding. Zero auxiliary variables.

2B) Write the name of the encoding for this constraint that needs the fewest clauses. How many?

Solution: Both the Ladder and the Heule 3 encodings need only $3n$ clauses.

2C) We call such an AMO encoding *arc-consistent for unit propagation* if, when one of $l_1 \dots l_n$ becomes true, the SAT solver's unit propagation mechanism will set the other literals to false. This is a good property. Does the encoding you gave as a answer for 2A fulfill it?

Solution: Yes, if a literal l_i becomes true, then for every other literal l_j there is a clause $\neg l_i \vee \neg l_j$ by which unit propagation will set l_j to false.

2D) Write all clauses needed to express the cardinality constraint $l_1 + \cdots + l_5 \geq 2$ without using any auxiliary variables (do not write any unnecessary clauses).

Solution: $l_1 \vee l_2 \vee l_3 \vee l_4$ $\quad$ $l_1 \vee l_2 \vee l_3 \vee l_5$ $\quad$ $l_1 \vee l_2 \vee l_4 \vee l_5$ $\quad$ $l_1 \vee l_3 \vee l_4 \vee l_5$ $\quad$ $l_2 \vee l_3 \vee l_4 \vee l_5$.

2E) Write all clauses needed to express the Pseudo-Boolean constraint $7x+4y+5z+3u+8v+9w \leq 16$ without using any auxiliary variables (do not write any unnecessary clauses).
Solution: $\neg w \vee \neg v$ $\quad \neg w \vee \neg x \vee \neg z$ $\quad \neg w \vee \neg x \vee \neg y$ $\quad \neg w \vee \neg x \vee \neg u$ $\quad \neg w \vee \neg z \vee \neg y$ $\quad \neg w \vee \neg z \vee \neg u$
$\neg v \vee \neg x \vee \neg z$ $\quad \neg v \vee \neg x \vee \neg y$ $\quad \neg v \vee \neg x \vee \neg u$ $\quad \neg v \vee \neg z \vee \neg y$ $\quad \neg x \vee \neg z \vee \neg y \vee \neg u$.

3) We need to plan the activities of a transportation company during a period of H hours. The company has T trucks, D drivers and there are N transportation tasks to be done, each one of which lasts one hour and needs one driver per truck.
Each task $i \in 1 \dots N$ needs K_i trucks, and has a list $L_i \subseteq \{1 \dots H\}$ of hours at which this task i can take place. For example, if $L_7 = \{3, 4, 8\}$ this means that task 7 can take place at hour 3, at hour 4 or at hour 8.
For each driver $d \in 1 \dots D$ there is a list of blockings $B_d \subseteq \{1 \dots H\}$ of hours at which driver D can *not* work.
3A) Explain how to use a SAT solver for planning this: for each task, when does it take place, and using which drivers. Clearly indicate which types of propositional variables you are using, and how many of each type, using the following format:
variables $t_{i,h}$ meaning "task i takes place at hour h"
for all tasks $i \in 1 \dots N$ and for all hours $h \in 1 \dots H$
Total: $N \cdot H$ variables.
Since H, D and N may be large, it is not allowed to use $O(H \cdot D \cdot N)$ variables (but using such a large number of clauses is fine). Hint: you many use several types of variables, for example one type with $N \cdot H$ variables and another one with $N \cdot D$.
Also clearly indicate which clauses you need, and how many of each type, and how many literals each type of clause has. If you use any AMO, cardinality or pseudo-Boolean constraints, it is not necessary to convert these into CNF.
Solution:
Variables $t_{i,h}$ meaning "task i takes place at hour h"
for all tasks $i \in 1 \dots N$ and for all hours $h \in 1 \dots H$
Total: $N \cdot H$ variables.
Variables $dr_{i,d}$ meaning "task i has driver d among its drivers"
for all tasks $i \in 1 \dots N$ and for all drivers $d \in 1 \dots D$
Total: $N \cdot D$ variables.

Clauses to express that each task i is at exactly one hour from its list L_i:
for each i, if $L_i = \{h_1, \dots h_k\}$, one cardinality constraint $t_{i,h_1} + \dots + t_{i,h_k} = 1$ (or one AMO+ALO).
Clauses to express that no driver d works on two different tasks i and j at the same hour h:
for each driver d and hour h and for each pair of tasks i, j such that $h \in L_i$ and $h \in L_j$,
a four-literal clause $\neg dr_{i,d} \vee \neg dr_{j,d} \vee \neg t_{i,h} \vee \neg t_{j,h}$.
Clauses to express that each task i needs K_i drivers:
one cardinality constraint $dr_{i,1} + \dots + dr_{i,D} = K_i$ per task (here $\geq$ would suffice instead of $=$).
Clauses to express that at no time point we are using more than T trucks:
for each hour h, a pseudo-Boolean constraint $K_1\, t_{1,h} + \ldots + K_N\, t_{N,h} \leq T$.
Clauses to express that a driver d cannot work at hour h:
for each task i, a two-literal clause $\neg t_{i,h} \vee \neg dr_{i,d}$
i.e., for each such a blocking of a driver d at hour h there are N two-literal clauses.

3B) Extend your solution to take into account that no driver can work more than 50 hours in total during the whole period of H hours. Hint: this may require another type of variables.
Solution:
Clauses to express that no driver works more than 50 hours:
for each driver d, one cardinality constraint $dr_{i,d} + \dots + dr_{N,d} \leq 50$