Logical reasoning and programming, lab session 4

(October 13, 2025)

4.1 How many symmetries does your formulation of PHP_n^{n+1} have?

4.2 We can define the lexicographic order on two bit vectors $x_1, \dots, x_n$ and $y_1, \dots, y_n$, denoted $x_1 \dots x_n \leq_{lex} y_1 \dots y_n$, as follows

$$\bigwedge_{i=1}^{n} ((\overline{x_i} \vee y_i \vee \overline{a_{i-1}}) \wedge (\overline{x_i} \vee a_i \vee \overline{a_{i-1}}) \wedge (y_i \vee a_i \vee \overline{a_{i-1}})),$$

where $\overline{a_0}$ is always false, using new auxiliary variables $a_0, a_1, \dots, a_{n-1}, a_n$.

(a) What is the purpose of auxiliary variables?

Hint: When is it necessary to satisfy $x_i \leq y_i$?

(b) Why is $\overline{a_0}$ always false and hence useless?

(c) Why can we replace $(\overline{x_n} \vee y_n \vee \overline{a_{n-1}}) \wedge (\overline{x_n} \vee a_n \vee \overline{a_{n-1}}) \wedge (y_n \vee a_n \vee \overline{a_{n-1}})$ just by $(\overline{x_n} \vee y_n \vee \overline{a_{n-1}})$? Hence we need only $3n - 2$ clauses and $n - 1$ auxiliary variables (a_n is also useless).

(d) How does the meaning of the formula change if you replace $(\overline{x_n} \vee y_n \vee \overline{a_{n-1}})$ by $(\overline{x_n} \vee \overline{a_{n-1}}) \wedge (y_n \vee \overline{a_{n-1}})$?

4.3 How can we exploit the lexicographic order to decrease the number of symmetries in PHP_n^{n+1}?

Hint: Order hole-occupancy or pigeon-occupancy vectors.

4.4 A very nice symmetry breaker for PHP_n^{n+1} is based on columnwise symmetry, namely we can add the following clauses

$$p_{i(i+1)} \vee \overline{p_{ij}}$$

for $1 \leq i < j \leq n$, where p_{kl} means that pigeon k is in hole l, for $1 \leq k \leq n + 1$ and $1 \leq l \leq n$. Why?

4.5 Try PicoSAT/pycosat and PySAT on PHP_n^{n+1} with various symmetry breakers.

4.6 Symmetry breaking and PHP_n^{n+1}, for further details see Knuth's TAOCP on satisfiability or slides Symmetry in SAT: an overview.

4.7 Try BreakID.

4.8 For some experimental results on graph coloring (discussed during the previous lab session), you can consult SAT Encoding of Partial Ordering Models for Graph Coloring Problems from SAT 2024.

4.9 Do you know how to efficiently express

$$p_1 + p_2 + \cdots + p_{100} \leq 99?$$

4.10 Is it possible to replace

$$p_1 + \cdots + p_{1024} \leq 1$$

by

$$p_1 + \cdots + p_{512} + x \leq 1 \text{ and } p_{513} + \cdots + p_{1024} + \overline{x} \leq 1?$$

If so, is it equivalent, or equisatisfiable?

4.11 There are various encodings of cardinality constraints, discuss sequential counter and bitwise encodings. You can find further examples in this presentation, this presentation, or in PySAT.

4.12 For an example of a cardinality constraint using if-then-else and BDDs check this presentation.

4.13 Check the API documentation of PySAT. There are various useful things, for example, `IDPool`, `clausify`, `enum_models`, `get_core`. For using assumptions in PySAT, see Module Description.

4.14 Check some examples in PySAT.

4.15 Formulate the software package upgradability as a MaxSAT problem, see this tutorial.

4.16 Check using implicit hitting set for MaxSAT in this tutorial.

4.17 Try MaxSAT in PySAT, check WCNF.

4.18 Try CBMC on this example. You can also try this program. For details, see these lecture notes.

4.19 We have a language that contains only one binary predicate symbol $\in$ and we have an interpretation $\mathcal{M} = (D, i)$ such that $D = \{a, b, c, d\}$ and $i(\in)$ is given by the following diagram:

$$\begin{tikzcd}
c \arrow[loop above] \arrow[r] & d \\
a \arrow[u] & b \arrow[ul]
\end{tikzcd}$$

Meaning that $x \in y$ iff there is an arrow from x to y. Decide whether the following formulae are valid in $\mathcal{M}$:

(a) $\exists X \forall Y (\neg(Y \in X))$,

(b) $\exists X \forall Y (Y \in X)$,

(c) $\exists X \forall Y (Y \in X \leftrightarrow Y \in Y)$,

(d) $\exists X \forall Y (Y \in X \leftrightarrow \neg(Y \in Y))$.