Software Verification

Wednesday, January 4th 2017, 3 hours

This assignment contains three independent parts: the first part deals with bounded model-checking, the second part is about abstract interpretation, and the last part addresses the termination problem for Petri nets.

All documents are authorized during the examination

1 Bounded Model-Checking (8pts)

This section will browse a few concepts that have been seen in the first part of the course (Bounded Model-Checking).

1.1 Binary Decision Diagrams

Question 1 *Given* $A = (x \wedge y) \vee (x \wedge t \wedge \neg z) \vee (y \wedge z)$ *and* $B = (\neg x \vee y) \wedge (\neg x \vee z) \wedge (\neg t \vee y)$*, two propositional logic formula, build the resulting binary decision diagrams for* A*,* B *and* $(A \vee B) \wedge (\neg A \wedge B)$ *(order on variables is (top)* $x > y > z > t$ *(bottom)).*

1.2 Slitherlink: "*Look like the innocent flower, but be the serpent under't*"

Slitherlink is a puzzle game similar to Sudoku which is played on a square lattice of dots. Some of the cells formed by the dots have numbers inside them. The goal of the game is to draw a single cycle without crossings along these edges, where the numbers in the cells determine how many of the four surrounding edges have to be connected (see Figure 1).

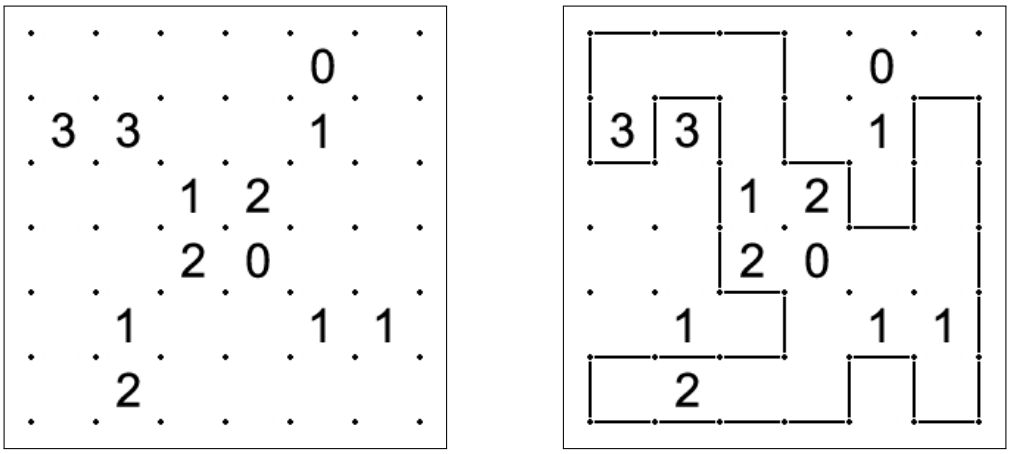


Figure 1: An example of Slitherlink grid: Unsolved (left), Solved (right).

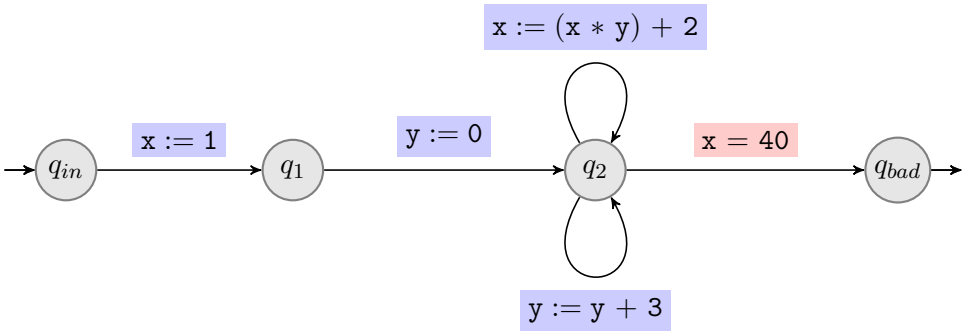
Question 2 *In fact, the "*one and only one cycle*" is a difficult constraint to encode. So, the point is to modelize this game with propositional logic without the constraint of having a unique cycle (you may have several independant cycles in your solution). Describe the variables that you use and the way you will store a problem instance. Then, explain the constraints that will encode the rules of the game (no crossing, satisfying cell's numbers).*

Hint (Question 2) *Encoding that you have only cycle(s) (one or several) can be done by adding constraints over the degree of each dot (the number of edges attached to dots).*

Question 3 *Think about a way to encode the "*one and only one cycle*" property. It might be totally inefficient and produce an exponential size formula, but do not care about efficiency!*

2 Range Analysis and Simple Congruence Analysis (8pts)

In this section, we analyze by abstract interpretation the control-flow automaton depicted below. This control-flow automaton has two variables `x` and `y`, both ranging over the set $\mathbb{Z}$ of integers. The initial location is q_{in} and the bad location is q_{bad}. Recall that '`:=`' denotes an assignment and that '`=`' denotes a condition. In all questions of this section, round-robin iteration shall use the following order on locations: $q_{in}, q_1, q_2, q_{bad}$.



2.1 Range Analysis

The questions of this subsection are concerned with range analysis of the control-flow automaton depicted above. This analysis was presented in the course.

Question 4 *Apply the round-robin algorithm* without *widening and* without *narrowing. Stop the round-robin iteration when the abstract value obtained for q_{bad} is distinct from $\bot$.*

Question 5 *Apply the round-robin algorithm* with *widening and* without *narrowing.*

2.2 Simple Congruence Analysis

Let us first introduce a few notations. The sum $X+Y$ and product $X \cdot Y$ of two subsets X and Y of $\mathbb{Z}$ are, respectively, the sets $\{x+y \mid x \in X \wedge y \in Y\}$ and $\{x \cdot y \mid x \in X \wedge y \in Y\}$. We abuse notation to reduce clutter by dropping dots in products and curly braces around singletons. For instance, $a + b\mathbb{Z}$ stands for $\{a\} + (\{b\} \cdot \mathbb{Z})$. For every integers $a, b \in \mathbb{Z}$ such

that $b > 0$, the *residue of a modulo b*, written $a \bmod b$, is the unique integer $r \in \{0, \ldots, b-1\}$ such that $a \in (r + b\mathbb{Z})$. The following facts, which can be used without proof, directly follow from the definition.

Fact 1 *For every integers $a, b \in \mathbb{Z}$ such that $b > 0$, it holds that $a \in (a \bmod b) + b\mathbb{Z}$.*

Fact 2 *For every integers $a, b, r \in \mathbb{Z}$ such that $b > r \geq 0$, if $a \in (r + b\mathbb{Z})$ then $r = (a \bmod b)$.*

For the remainder of this subsection, we consider a fixed positive integer $p > 0$. Let A denote the set of all subsets of $\{0, \ldots, p-1\}$. Observe that $(A, \subseteq)$ is a complete lattice. We define the functions $\alpha : \mathcal{P}(\mathbb{Z}) \to A$ and $\gamma : A \to \mathcal{P}(\mathbb{Z})$ as follows:

$$\alpha(X) \;=\; \{x \bmod p \mid x \in X\} \qquad\qquad \gamma(M) \;=\; M + p\mathbb{Z}$$

for every $X \subseteq \mathbb{Z}$ and $M \subseteq \{0, \ldots, p-1\}$.

Question 6 *Prove that $(\mathcal{P}(\mathbb{Z}), \subseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (A, \subseteq)$ is a Galois connection.*

By *simple congruence analysis*, we mean the abstract interpretation based on the above Galois connection. Observe that a widening is not required here since the abstract lattice $(A, \subseteq)$ is finite. The following question instantiates simple congruence analysis for $p = 6$.

Question 7 *Apply the round-robin algorithm to perform simple congruence analysis for $p = 6$ of the control-flow automaton depicted above.*

3 Petri Nets Termination Problem (10pts)

In this section $(P, T, \text{input}, \text{output})$ denotes a *Petri net* where P is a finite set of *places*, T is a finite set of *transitions*, and $\text{input}, \text{output} : T \to P \to \mathbb{N}$ are the input and output functions of the Petri net. We recall that a *marking* is a function $m : P \to \mathbb{N}$, and the Petri net semantics is defined by introducing for each transition $t \in T$ the binary relation $\xrightarrow{t}$ over the markings defined as follows:

$$x \xrightarrow{t} y \iff \forall p \in P \;:\; x(p) \geq \text{input}(t)(p) \;\wedge\; y(p) = x(p) - \text{input}(t)(p) + \text{output}(t)(p)$$

We associate to every word $\sigma = t_1 \ldots t_k$ of transitions $t_1, \ldots, t_k$ in T the binary relation $\xrightarrow{\sigma}$ over the markings defined by $x \xrightarrow{\sigma} y$ if there exists a sequence $m_0, \ldots, m_k$ of markings such that:

$$x = m_0 \xrightarrow{t_1} m_1 \cdots \xrightarrow{t_k} m_k = y$$

The *reachability relation* is the binary relation $\xrightarrow{*}$ over the markings defined by $x \xrightarrow{*} y$ if there exists a word σ of transitions such that $x \xrightarrow{\sigma} y$.

We say that a Petri net is *non-terminating* from an initial marking m_{init} if there exists an infinite sequence $t_1, t_2, \ldots$ of transitions and an infinite sequence $m_0, m_1, m_2, \ldots$ of markings such that:

$$m_{\text{init}} = m_0 \xrightarrow{t_1} m_1 \xrightarrow{t_2} m_2 \cdots$$

Otherwise, we say that the Petri net is *terminating*. The termination problem consists in deciding if a Petri net is terminating or non-terminating from an initial marking.

The sum $x+y$ of two markings x, y is the marking defined by $(x+y)(p) = x(p)+y(p)$ for every place p in P.

Question 8 *Prove that $x + m \xrightarrow{*} y + m$ for every markings x, y, m such that $x \xrightarrow{*} y$.*

We introduce the partial order $\leq$ over the markings defined by $x \leq y$ if $x(p) \leq y(p)$ for every place p in P.

Question 9 *Assume that $m_{init} \xrightarrow{*} x \xrightarrow{*} y$ for some markings m_{init}, x, y such that $x \leq y$. Prove that the Petri net is non-terminating from m_{init}.*

Question 10 *Prove by induction over the cardinal of the set of places P that every infinite sequence $m_0, m_1, m_2, \ldots$ of markings contains an infinite non-decreasing for $\leq$ subsequence, i.e. $m_{i_0} \leq m_{i_1} \leq m_{i_2} \leq \cdots$ where $i_0 < i_1 < i_2 < \cdots$. This result is called* Dickson's Lemma.

Question 11 *Prove that if the Petri net is non-terminating from the initial marking m_{init}, then there exists two markings x, y such that $m_{init} \xrightarrow{*} x \xrightarrow{*} y$ and $x \leq y$.*

Question 12 *Explain in few lines how to decide the Petri net termination problem.*