Software Verification

Monday, January 9th 2023, 3 hours

This assignment contains three independent parts: the first part deals with bounded model-checking, the second part is about trivector abstraction, and the last part addresses abstract interpretation.

All documents are authorized during the examination
Answers can be written in French

1 Bounded Model-Checking (10pts)

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

1.1 Principles

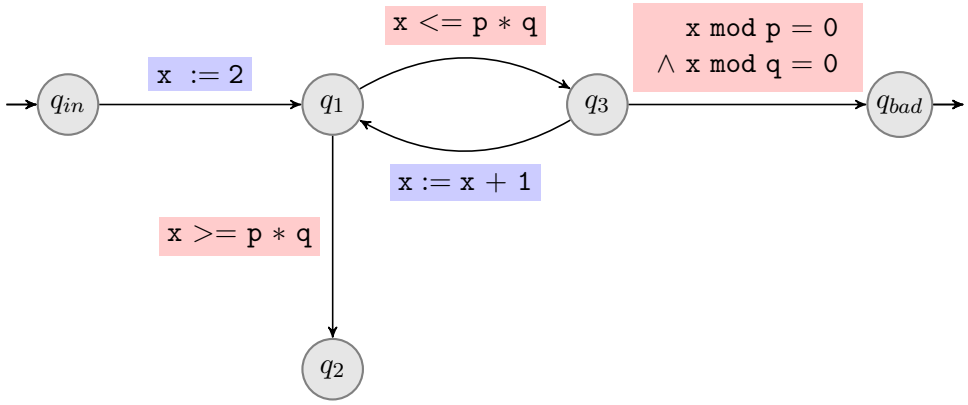
We start with some questions to check your understandings of the Bounded Model-Checking algorithms. We recall that given a transition system $\mathcal{T}$, the Bounded Model-Checking at depth n explores all executions of length at most n and checks if one of them reaches an undesirable state (such an execution is called *faulty*).

Question 1 *A Bounded Model-Checking search can return three answers: Counter-example found, No counter-example found (exhaustive), No counter-example found (non-exhaustive).*

For each possible answer, explain what the answer means about the presence of a faulty execution, its size, and the size of all executions of the model.

Question 2 *We have seen in the course two algorithms: Depth-First Search, and Global. Recall briefly how the executions are explored in the two algorithms and what guarantee they give on the faulty execution they return if they return one.*

1.2 Executing on an example



Observe that in the system depicted above, we have allowed to test two conditions at the same time (to shorten the model for making the paper exploration easier). You will of course use formulæ that do the two at the same time.

The state q_2 is considered to be the ending state of the program. q_{bad} is of course the error state that we want to test if it is reachable or not.

p and q are constants in the whole program (they can be seen as arguments). In all this exercise, we will suppose they are strictly bigger than 1 (as otherwise, the system makes little sense).

Question 3 *We consider in this question that $p = 2$ and $q = 3$.*

Apply the bounded model-checking algorithm to the previous system up to bound 6: draw the exploration tree displaying the formulæ added (only the added one not the complete formula) at each step in the Depth-First Search algorithm, highlight the unreachable nodes, and give the result of the algorithm.

Question 4 *Is there a bound on which the algorithm would give a different answer (still with $p = 2$ and $q = 3$)? Justify (either explain why the algorithm will always answer correct, or give a failing execution).*

In case there is a faulty execution, propose a modifications to the tests starting in q_1 that will ensure the system is correct (still with $p = 2$ and $q = 3$)

Question 5 *We now consider arbitrary values for p and q. Is the system correct in that case (taking into account your previous correction)?*

If not, for which values of p and q is the system incorrect, and for which one is it correct? Justify your answer.

1.3 The Bounded Variable Case

In this section, we suppose that we work on a system with states Q and n variables on which it has already be proven that the values of the variables are at any time between 0 and m, for some integer m.

Question 6 *Prove that in that setting, the Bounded Model-Checking is complete, i.e. there exists a bound k for which if the algorithm has not found a counter-example, we know there is none, even for longer executions (even if the algorithm answered Non-exhaustive).*

You will precise the bound for which we can be certain the algorithm is complete, by expressing it with respect to $|Q|$, n and m.

What is thus the worst-case complexity of BMC in this case?

Question 7 *The goal of this question is to explore a trade-off of on time and space complexity we can make, if we throw the Bounded Model-Checking to the trash and try to directly solve model-checking.*

We consider we have access to a hashtable structure that can associate to every configuration a boolean, and have (amortised) constant cost of accessing and modifying values (which is a realistic assumption: it is the case for the module Hashtbl in OCaml). Suppose that on this structure, you have two functions : `add(table,key,value)` *that associates*

`value` to `key` in `table`, and `find(table,key)` that returns the value associated to `key` in `table` (and returns false if the key is absent).

With that in mind, propose an algorithm for Model-Checking that uses a hashtable, and no SMT-solver, and explores the descendants of any configuration at most once. Give a detailed pseudo-code of it, where you suppose that (as in the project) you can access every useful element of your automaton (like the successors of a node), and you can iterate over arbitrary structures (like lists or sets). This pseudo-code can be pseudo-imperative or pseudo-OCaml (as you prefer).

What is the time complexity of your algorithm (in the number of configurations, that you will express with respect to $|Q|$, n and m)?

What is its space complexity?

If we suppose (as it is reasonable) that m is given in binary, what is the complexity of this algorithm (both time and space) with respect to its input size? Is it thus a reasonable algorithm (your answer should be nuanced)?

2 Trivector Abstraction (8pts)

We consider the control-flow automaton depicted in Figure 1 where variables $\{\mathtt{x}, \mathtt{b}\}$ are valued over the integers. We are interested in proving that whatever the initial valuation of $\mathtt{x}$ and $\mathtt{b}$, the location q_{bad} is not reachable from q_{in}.

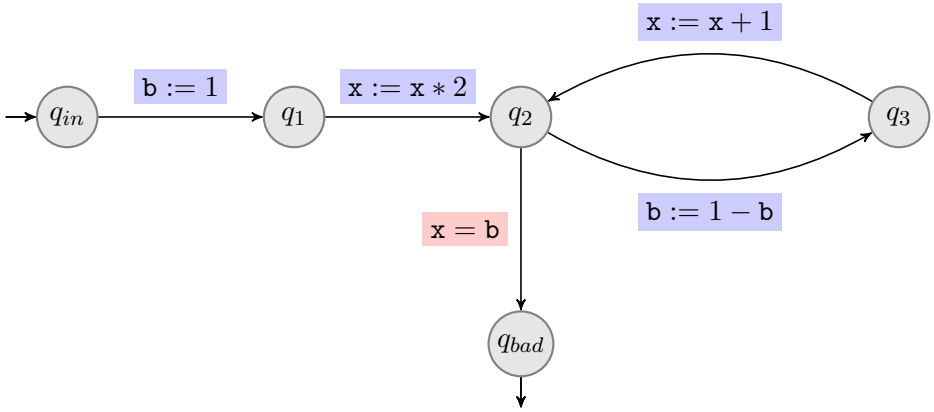


Figure 1: A control-flow automaton.

Question 8 *Let us first consider the control-flow automaton depicted in Figure 1 but with the assignment* $\mathtt{x} := \mathtt{x} + 1$ *replaced by* $\mathtt{x} := \mathtt{x} + \mathtt{b}$. *Provide a concrete trace explaining why the location* q_{bad} *is reachable from* q_{in} *in that case.*

Now, we come back to the control-flow automaton depicted Figure 1.

Question 9 *Provide an unsatisfiable logical formula explaining why the trace* $q_{in} \xrightarrow{\mathtt{b:=1}} q_1 \xrightarrow{\mathtt{x:=x*2}} q_2 \xrightarrow{\mathtt{x=b}} q_{bad}$ *is spurious.*

Question 10 *Provide a path invariant* $(\phi_0, \phi_1, \phi_2, \phi_3)$ *for the trace of the previous question based on a weakest precondition computation. Justify that* ϕ_0 *is equivalent to true.*

Question 11 *We consider the predicates* $p_1 = (\mathtt{b} = 1)$ *and* $p_2 = (\mathtt{x} - \mathtt{b}$ *is even). Provide the (reachable part) of the abstract transition system for the trivector abstraction starting from the node* $(q_{in}, (\star, \star))$.

Question 12 *Deduce for each location* q *a predicate* ϕ_q *in such a way* $\phi_{q_{in}}$ *is true,* $\phi_{q_{bad}}$ *is false, and such that for every transition* $p \xrightarrow{op} q$, *we have:*

$$\mathrm{post}_{op}([\![\phi_p]\!]) \subseteq [\![\phi_q]\!]$$

3 Abstract Interpretation (12pts)

The main objective of this section is to show that abstract interpretation, which was presented using Galois connections in the course, can also be phrased in terms of so-called closure operators.

3.1 Abstract interpretation with closure operators

We start by recalling some basic notions. A *complete lattice* is a partially-ordered set $(L, \sqsubseteq)$ such that every subset $X \subseteq L$ admits a greatest lower bound and a least upper bound. A function $f : L \to L$ on a complete lattice $(L, \sqsubseteq)$ is called

- *monotonic* if $(x \sqsubseteq y \implies f(x) \sqsubseteq f(y))$ for every $x, y \in L$,
- *reductive* if $f(x) \sqsubseteq x$ for every $x \in L$,
- *extensive* if $f(x) \sqsupseteq x$ for every $x \in L$,
- *idempotent* if $f(x) = f(f(x))$ for every $x \in L$.

We also recall the characterization of Galois connections that was given in the course. Given two complete lattices $(C, \sqsubseteq)$ and $(A, \preceq)$, a pair of functions $\alpha : C \to A$ and $\gamma : A \to C$ forms a *Galois connection*, written $(C, \sqsubseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (A, \preceq)$, if, and only if, α and γ are monotonic, $\alpha \circ \gamma$ is reductive, and $\gamma \circ \alpha$ is extensive.

We now introduce the notion of closure operator and show that every Galois connection induces a closure operator. A *closure operator* on a complete lattice $(L, \sqsubseteq)$ is a function $\sigma : L \to L$ that is monotonic, extensive and idempotent.

Question 13 *Assume that $(C, \sqsubseteq)$ and $(A, \preceq)$ are two complete lattices. Prove that, for every Galois connection $(C, \sqsubseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (A, \preceq)$, the function $\gamma \circ \alpha$ is a closure operator.*

The proof that every closure operator induces a Galois connection will require more work. For the remainder of this subsection, we fix a complete lattice $(C, \sqsubseteq)$ and a closure operator σ on C. The greatest lower bound and least upper bound of $(C, \sqsubseteq)$ are written $\sqcap$ and $\sqcup$, respectively.

An element $c \in C$ is called *closed* if $\sigma(c) = c$. Let A denote the set of all closed elements of C. Observe that, by idempotence, we have $A = \{\sigma(c) \mid c \in C\}$. Let $\preceq$ denote the restriction to A of the partial order $\sqsubseteq$ on C. Formally, $\preceq$ is the intersection of $\sqsubseteq$ and $(A \times A)$. Observe that $\preceq$ is a partial order on A. Our first objective is to show that the partially-ordered set $(A, \preceq)$ is a complete lattice.

Question 14 *Let $X \subseteq A$. Prove that $\sigma(\sqcup X)$ is the least upper bound of X in $(A, \preceq)$.*

Question 15 *Let $X \subseteq A$. Prove that $\sqcap X$ is closed (i.e., $\sigma(\sqcap X) = \sqcap X$). Deduce that $\sqcap X$ is the greatest lower bound of X in $(A, \preceq)$.*

We have shown that $(A, \preceq)$ is a complete lattice with greatest lower bound $\wedge$ and least upper bound $\vee$ verifying $\wedge X = \sqcap X$ and $\vee X = \sigma(\sqcup X)$, for every subset $X \subseteq A$.

Question 16 *Prove that $(C, \sqsubseteq) \underset{\sigma}{\overset{id}{\leftrightarrows}} (A, \preceq)$ is a Galois connection.*[1]

[1] The function $id : A \to C$ is the *identity* function (i.e., $id(a) = a$ for all $a \in A$).

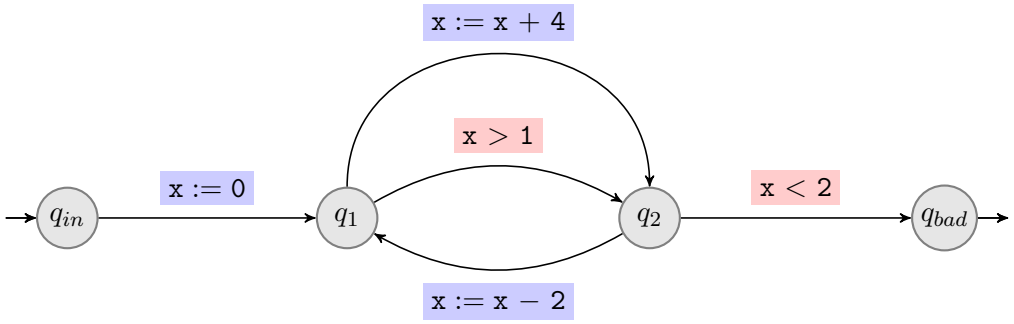


Figure 2: Control-flow automaton of Question 19.

3.2 Application to program analysis

We now apply the results of the previous subsection to program analysis by abstract interpretation. Assume a fixed parameter $t \in \mathbb{N}$ which shall serve as a threshold. Consider the function $\theta : \mathcal{P}(\mathbb{Z}) \to \mathcal{P}(\mathbb{Z})$ defined by $\theta(S) = S_- \cup S_0 \cup S_+$ where $S_0 = S \cap [-t, t]$ and

$$S_- = \begin{cases}]-\infty, -t[& \text{if } \exists s \in S : s < -t \\ \emptyset & \text{otherwise} \end{cases} \qquad S_+ = \begin{cases}]t, +\infty[& \text{if } \exists s \in S : s > t \\ \emptyset & \text{otherwise} \end{cases}$$

For instance, assuming that the threshold t is 5, we have $\theta(\{-1, 2, 4\}) = \{-1, 2, 4\}$ and $\theta(\{-7, 2, 4\}) =]-\infty, -5[\cup \{2, 4\}$.

Question 17 *Prove that θ is a closure operator on the complete lattice $(\mathcal{P}(\mathbb{Z}), \subseteq)$.*

As in subsection 3.1, we let $A = \{\theta(S) \mid S \subseteq \mathbb{Z}\}$ denote the set of closed elements of $\mathcal{P}(\mathbb{Z})$, and we write $\preceq$ the restriction to A of the partial order $\subseteq$ on $\mathcal{P}(\mathbb{Z})$. The results of subsection 3.1 ensure that $(A, \preceq)$ is a complete lattice and that $(\mathcal{P}(\mathbb{Z}), \subseteq) \underset{\theta}{\overset{id}{\leftrightarrows}} (A, \preceq)$ is a Galois connection.

Question 18 *Does the abstract lattice $(A, \preceq)$ satisfy the ascending chain condition? Justify your answer.*

Consider the control-flow automaton depicted in Figure 2. This control-flow automaton has a single variable `x`, which ranges over integers. The initial location is q_{in} and the bad location is q_{bad}. Like in the course, an analysis will be called *successful* when the abstract value obtained for q_{bad} has an empty concretization. Round-robin iteration shall use the following order on locations: $q_{in}, q_1, q_2, q_{bad}$.

Question 19 *Apply the round-robin algorithm to analyze the control-flow automaton depicted in Figure 2 using the Galois connection $(\mathcal{P}(\mathbb{Z}), \subseteq) \underset{\theta}{\overset{id}{\leftrightarrows}} (A, \preceq)$, assuming that the threshold t is 5. Is the analysis successful?*