Q&A

2025/02/28

Reference: https://www.sosy-lab.org/research/pub/2018-HBMC.Combining_Model_Checking_and_Data-Flow_Analysis.pdf

1 Questions and Answers

1. How does an observer automaton look like for the different LTL operators?

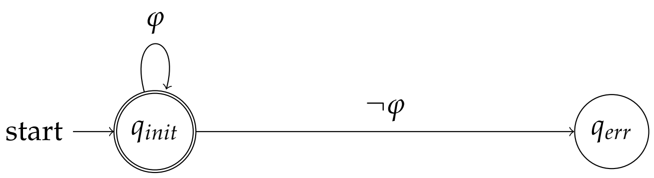

Figure 1: $\square\phi$ (always φ)

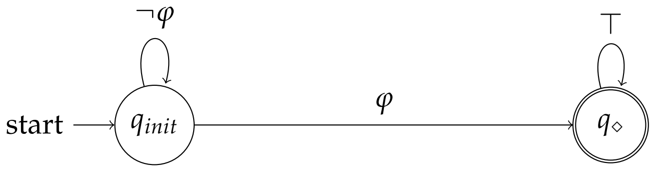

Figure 2: $\diamond\varphi$ (finally φ)

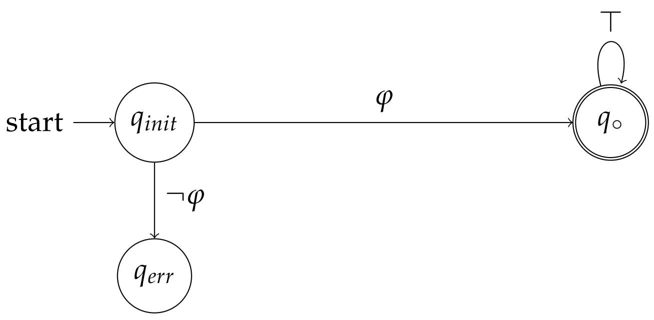

Figure 3: $\circ\phi$ (next ϕ)

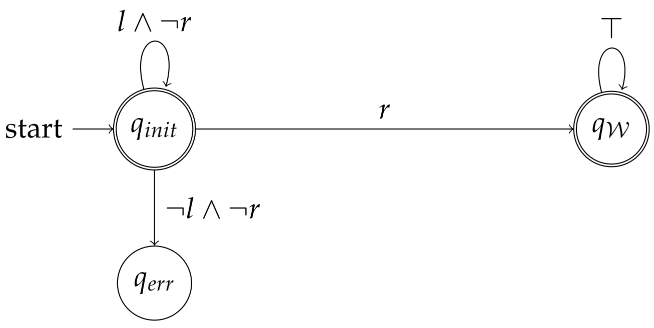

Figure 4: $l \; \mathcal{W} \; r$ (l weak until r)

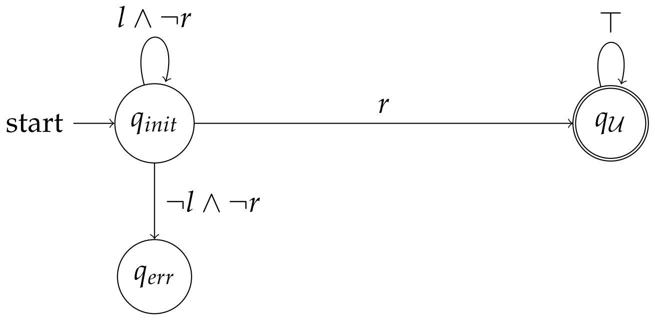

Figure 5: $l \,\mathcal{U}\, r$ (l until r)

2. When does the $\mathsf{stop_{sep}}$ operator return true with abstract domain $\mathbb{P}$?

   The operator $\mathsf{stop_{sep}}(e, \mathsf{reached})$ returns true if $\exists e' \in \mathsf{reached} : e \sqsubseteq e'$. For $\mathbb{P}$ we define $\sqsubseteq$ with $\supseteq$, meaning that we stop whenever we have an element in the reached set, that is a subset of the current element e. Why is this sound? Because we have already explored a superset of abstract paths that this state would allow.

3. Is reachability analysis glorified dead code detection?

   Yes. However, it is not easy to check a magnitude of existing real states. If we want to verify a function with input parameter y, then we would have to manually check the unreachability of every concrete assignment. Assume $\mathsf{int} = \mathbb{Z}$, then we already have infinitely many states to check. With abstraction we can track constraints over y, helping us to bundle infinitely many states (e.g., with $y < 0$) in one abstract state allowing us to decide the reachability of certain locations for simple programs. Additionally, it is crucial that all asserts hold in a program. Therefore, it makes sense to use reachability analysis to check for possible violations.

2 Questions

The below questions are supposed to support you in exam preparation. They are not meant to be complete (i.e., they do **not** represent all content that you have to know).

Software Verification

1. Define the notion of "Software Verification".
2. What is the difference between formal verification and testing?

Lattices

1. Define lattice in words.
2. Define a semi-lattice.
3. What is the meaning of the individual components of a lattice?

CPA Algorithm

1. What is the purpose of reachability analyses?
2. What is the difference between model checking and data-flow analysis?
3. How does the CPA algorithm differ from model checking?
4. Name all components of a CPA and state the purpose of each.

Constant-propagation Analysis, Reaching Definitions

1. What information does the constant-propagation analysis track?
2. What information does the reaching-definitions analysis track?
3. Let us assume our constant-propagation analysis does not use $merge^{join}$, but $merge^{sep}$. How does its behavior change?

Bounded Model Checking

1. Is bounded model checking more expressive than constant-propagation analysis?
2. How does bounded model checking work?
3. Is it possible to prove a program correct with regards to a certain property, with bounded model checking?

Predicate Abstraction

1. What is the difference between bounded model checking and predicate abstraction?

Observer Analysis

1. What do we use the observer analysis for?
2. Is it possible to apply the CPA algorithm with more than a single observer analysis at the same time? If so, how?
3. Can observer analyses describe any program property?