PROVING TEMPORAL PROPERTIES BY ABSTRACT INTERPRETATION MASTER THESIS PROJECT DESCRIPTION

Samuel Ueltschi
Supervisor: Caterina Urban

ETH Zürich
Zürich, Switzerland
March 2017

Introduction

Temporal logic is a widely accepted language for the specification of the intended behavior of various systems (operating systems, embedded systems, network communication protocols, etc.). Powerful tools have emerged over the years for proving temporal logic properties of programs. Useful properties that temporal logic allows to express are safety properties like partial correctness (i.e., the program does not produce the wrong answer), mutual exclusion (i.e., two processes are not in their critical section at the same time), and deadlock-freedom (i.e., the program does not reach a deadlock state), and liveness properties like termination (i.e., the program eventually terminates), and starvation freedom (i.e., a process eventually receives service).

Abstract interpretation is a general theory for approximating the semantics of a program that was developed by Patrick Cousot and Radhia Cousot in the late 1970s. The theory of abstract interpretation has been recently used to prove temporal logic properties, specifically guarantee (something good happens at least once) and recurrence properties (something good happens infinitely often) [1].

Liveness properties such as guarantee and recurrence properties can be expressed using computation tree logic (CTL). Guarantee properties are expressed by the CTL formula $\forall\Diamond\phi$ (for all paths there will be a state that satisfies ϕ), recurrence properties by $\forall\Box\forall\Diamond\phi$ (for all paths it will always be true that $\forall\Diamond\phi$ holds). Consider the example program in listing 1, there the guarantee property $\forall\Diamond(x = 5)$ and the recurrence property $\forall\Box\forall\Diamond(x = -10)$ can be shown to hold.

```
x = 0;
while(x < 10) x++;
while(true) x = -x;
```

Listing 1. Example Program

Proving temporal properties by abstract interpretation requires the definition of an abstract interpretation on the maximal trace semantics of a program. In case of guarantee properties $\forall\Diamond\phi$ this is done by defining the guarantee semantics $\tau_g[S] \in \Sigma \rightharpoonup \mathbb{O}$ that assign each program state an upper bound on the number of program steps until the guarantee property is satisfied. The guarantee semantics $\tau_g[S]$ is usually not computable. However, it can be approximated by the abstract domain of piecewise-defined ranking functions [2]. The semantics for recurrence properties $\forall\Box\forall\Diamond\phi$ is defined based on the definition of guarantee semantics.

The previous work on guarantee and recurrence properties demonstrates how atomic CTL formulae (e.g. $\forall\Diamond\phi$) can be approximated with abstract interpretation, and how the combination of atomic CTL formulae (e.g. $\forall\Box\forall\Diamond\phi$) can be defined by composing the semantics of the atomic operators. The goal of this thesis is to extend the existing techniques to support more atomic CTL formulae and to come up with ways to combine them to form more complex CTL formulae. Ideally, a significant subset that corresponds to a universal fragment of CTL (∀CTL) can be supported. In addition to that, the devised approach should also be implemented and integrated within the static analyzer FuncTion[1] and then evaluated with a set of experiments.

1. CORE GOALS

The core goal of this thesis is to extend the existing framework to support the universal fragment of CTL, which is given here by the following inductive definition:

$$\Phi ::= \top \mid \bot \mid a \mid \neg a \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid \forall \bigcirc \Phi \mid \forall\Diamond\Phi \mid \forall\Box\Phi \mid \forall(\Phi_1 \; U \; \Phi_2)$$

This requires the following steps:

[1] https://www.di.ens.fr/ urban/FuncTion.html

- Defining the concrete semantics for each atomic operator ($\forall\bigcirc$, $\forall\Box$, $\forall\Diamond$, $\forall U$), boolean combinator ($\wedge$, $\vee$) and the nesting of the operators with each other. Ideally, the concrete semantics of the atomic operators follow a compositional approach that allows arbitrary nesting of formulae.

- Defining an abstraction of the concrete semantics in the domain of piecewise-defined ranking functions. This might also require an extension of the abstract domain to support all operators of $\forall$CTL.

- Implementing the devised approach in FuncTion followed by an evaluation.

So far this has only been done for the atomic operator $\forall\Diamond$ and the corresponding composition $\forall\Box\forall\Diamond$.

2. EXTENSIONS

2.1. Existential Quantifiers in CTL

One possible extensions for this thesis would be to investigate supporting existential quantifiers of CTL e.g formulae of the form $\exists\Diamond\phi$, $\exists\Box\phi$ or $\forall\Box\exists\Diamond\phi$. The formula $\forall\Box\exists\Diamond(start)$ for instance, states that from each point in the program it is always possible to go back to the start. Supporting existential quantifiers would require extending the existing framework to support atomic operators $\exists\Box$, $\exists\Diamond$, $\exists\bigcirc$ and $\exists U$. Hopefully these can then be combined with the existing operators of the universal fragment by using the same compositional approach.

2.2. Support LTL Properties

Another possible extension is generalizing the approach to support properties expressed in (a significant subset of) linear-time temporal logic (LTL). Supporting LTL formulae is challenging because CTL and LTL are not comparable i.e. there are statements that can only be expressed in LTL but not CTL and vice versa. One example for this is the LTL formula $\Diamond\Box\phi$ and CTL formula $\forall\Diamond\forall\Box\phi$. Intuitively these may seem equivalent, however one can show that there are transition systems that satisfy the LTL but not the CTL formula [3]. Furthermore, LTL is not as easily composable as CTL. The semantic of CTL is defined in terms of states which makes it possible to take a set of states that satisfy one CTL formula as input to another atomic CTL operator. In LTL however, the semantic is defined in terms of traces starting from an initial state. If one can show that a property holds for a set of traces, it is not obvious how that information can be reused to show that a more complex LTL formula holds.

3. REFERENCES

[1] Caterina Urban and Antoine Miné, "Proving Guarantee and Recurrence Temporal Properties by Abstract Interpretation," in *VMCAI*, 2015, pp. 190–208.

[2] Caterina Urban, "The Abstract Domain of Segmented Ranking Functions," in *SAS*, 2013, pp. 43–62.

[3] Christel Baier and Joost-Pieter Katoen, *Principles of Model Checking (Representation and Mind Series)*, The MIT Press, 2008.