Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Software Model Checking with a Combination of Explicit Values and Predicates

Bajkai Viktória Dorina I. évf, (MSc) mérnökinformatikus szakos hallgató
Konzulens: Hajdu Ákos tanársegéd, MIT
Kritikus rendszerek szakirány
Önálló laboratórium 1. összefoglaló
2018/19. I. félév

Formal verification techniques (e.g. model checking) provide a sound mathematical basis to prove the correct operation of a program by exhaustively analyzing all possible states and transitions. This is also the downside of such methods, making them too expensive computationally. Counterexample-guided abstraction refinement (CEGAR) is a widely used approach to overcome this limitation. CEGAR applies abstraction to hide certain details and to over-approximate the set of possible behaviors of the program. However, this does not only yield a smaller state space, but can also lead to spurious counterexamples. In such cases, the algorithm automatically refines the abstraction and repeats this process until a sufficient precision is found. CEGAR can work with different abstract domains, such as explicit values and predicates. The former works by tracking only a subset of the program variables, while the latter stores certain relationships and facts about them.

Product abstractions, which combine multiple abstract domains also emerged, since different abstract domains turned out to be suitable for different kinds of software. In this paper we present a product abstraction algorithm, which combines explicit values and predicate abstraction, exploiting the advantages of each domain. The key idea of our approach is that we always start with explicit values to avoid handling formulas, but switch to predicates if there are too many values for a variable.

We implemented this algorithm in Theta, an open source verification framework, which already includes a generic CEGAR loop and some basic abstract domains. We evaluate our new approach and the existing methods on several programs from different problem domains, including PLC models form CERN and C programs from the Competition on Software Verification (SV-Comp). Our results show that the new product abstraction algorithm successfully combines the advantages of two existing domains. Related work: The dynamic precision adjustment approach combines predicates and explicit values. The main difference is that it switches to predicates based on the whole state space, whereas we only consider the successors of a single state. Moreover, we allow successors to be enumerated if an expression cannot be evaluated, as opposed to the unknown values in dynamic precision adjustment.

Refinement selection chooses between the explicit and predicate domains based on various metrics for the refinement quality. Our approach always tries the explicit domain first, but switches to predicates if there are too many different values.