Information and Computation 252 (2017) 95–109

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

On the completeness of bounded model checking for threshold-based distributed algorithms: Reachability ☆

Igor Konnov, Helmut Veith, Josef Widder *

TU Wien (Vienna University of Technology), Austria

ARTICLE INFO

Article history:
Received 15 December 2014
Available online 2 March 2016

Keywords:
Model checking
Fault-tolerant distributed algorithms
Byzantine faults
Computational models

ABSTRACT

Counter abstraction is a powerful tool for parameterized model checking, if the number of local states of the concurrent processes is relatively small. In recent work, we introduced parametric interval counter abstraction that allowed us to verify the safety and liveness of threshold-based fault-tolerant distributed algorithms (FTDA). Due to state space explosion, applying this technique to distributed algorithms with hundreds of local states is challenging for state-of-the-art model checkers. In this paper, we demonstrate that reachability properties of FTDAs can be verified by bounded model checking. To ensure completeness, we need an upper bound on the distance between states. We show that the diameters of accelerated counter systems of FTDAs, and of their counter abstractions, have a quadratic upper bound in the number of local transitions. Our experiments show that the resulting bounds are sufficiently small to use bounded model checking for parameterized verification of reachability properties of several FTDAs, some of which have not been automatically verified before.

1. Introduction

A system that consists of concurrent anonymous (identical) processes can be modeled as a counter system: Instead of recording which process is in which local state, we record for each local state, how many processes are in this state. We have one counter per local state ℓ, denoted by $\kappa[\ell]$. Each counter is bounded by the number of processes. A step by a process that goes from local state ℓ to local state ℓ' is modeled by decrementing $\kappa[\ell]$ and incrementing $\kappa[\ell']$.

We consider a specific class of counter systems, namely those that are defined by *threshold automata*. The technical motivation to introduce threshold automata is to capture the relevant properties of fault-tolerant distributed algorithms (FTDAs). FTDAs are an important class of distributed algorithms that work even if a subset of the processes fails [26]. Typically, they are parameterized in the number of processes and the number of tolerated faulty processes. These numbers of processes are parameters of the verification problem. We show that the counter systems defined by threshold automata have a diameter whose bound is independent of the bound on the counters, but depends only on characteristics of the threshold automaton. This bound can be used for parameterized model checking of FTDAs, as we confirm by experimental evaluation.

☆ Supported by the Austrian Science Fund (FWF) through the National Research Network RiSE (S11403 and S11405) and project P27722 (PRAVDA), and by the Vienna Science and Technology Fund (WWTF) through project ICT15-103 (APALACHE) and grant PROSEED.

* Corresponding author.
E-mail address: widder@forsyte.at (J. Widder).

http://dx.doi.org/10.1016/j.ic.2016.03.006

Modeling FTDAs as counter systems defined by threshold automata A threshold automaton consists of rules that define the conditions and effects of changes to the local state of a process of a distributed algorithm. Conditions are *threshold guards* that compare the value of a shared variable to a linear combination of parameters, e.g., $x \geq n - t$, where x is a shared variable and n and t are parameters. This captures counting arguments which are used in FTDAs, e.g., a process takes a certain step only, if it has received a message from a majority of processes. To model this, we use the shared variable x as the number of processes that have sent a message, n as the number of processes in the system, and t as the assumed number of faulty processes. The condition $x \geq n - t$ then captures a majority under the resilience condition that $n > 2t$. Resilience conditions are standard assumptions for the correctness of an FTDA.[1] The effect of a rule of a threshold automaton is that a shared variable is increased, which naturally captures that a process has sent a message. As a process cannot undo the sending of a message, it is natural to consider threshold automata where shared variables are never decreased. In addition, we use shared variables to model the number of processes that have sent a specific message. To be able to do so, we have to restrict how often a process may send a specific message. In particular, to model the counting mechanism, we have to prevent that a process sends a message from within an infinite loop (or a loop where the number of iterations is unknown). We are thus led to consider threshold automata where rules that form cycles do not modify shared variables. While we add this restriction to derive our technical contribution, we do not consider it too limiting with respect to the application domain: Indeed, in all our case studies a process sends a given message at most once; this property appears natural if one considers distributed algorithms under the classic assumption of reliable communication.

Bounding the diameter For reachability it is not relevant whether we "move" processes one by one from local state ℓ to local state ℓ'. If several processes perform the same transition one after the other, we can model this as a single update on the counters: The sequence where b processes one after the other move from ℓ to ℓ' can be encoded as a single transition where $\kappa[\ell]$ is decreased by b and $\kappa[\ell']$ is increased by b. We call the value of b the *acceleration factor*. It may vary in a run depending on how many repetitions of the same transition should be captured. We call such runs of a counter system *accelerated*. The lengths of accelerated runs are the ones relevant for the diameter of the counter system.

Our central idea is that given a run that starts in configuration σ and ends in configuration σ', by swapping and accelerating transitions in that run, we can construct a run of bounded length that also starts in σ and ends in σ'. This bound then gives us the diameter. For deriving this bound, the main technical challenge comes from the interactions of shared variables and threshold guards. We address it with the following three ideas:

i. *Acceleration*. As discussed above.
ii. *Sorting*. Given an arbitrary run of a counter system, we can shorten it by changing the order of transitions such that there are possibly many consecutive transitions that can be merged according to (i), and the resulting run leads to the same configuration as the original run. However, as we have arithmetic threshold conditions, not all changes of the order result in allowed runs.
iii. *Segmentation*. We partition a run into segments, inside of which we can reorder the transitions; cf. (ii).

In combination, these three ideas enable us to prove the main theorem: *The diameter of a counter system is at most quadratic in the number of rules; more precisely, it is bounded by the product of the number of rules and the number of distinct threshold conditions.* In particular, the diameter is independent of the parameter values.

Using the bound for parameterized model checking Parameterized model checking is concerned with the verification of concurrent or distributed systems, where the number of processes is not a priori fixed, that is, a system is verified for all sizes [6]. In our case, the counter systems for all values of n and t that satisfy the resilience condition should be verified. A well-known parameterized model checking technique is to map all these counter systems to a *counter abstraction*, where the counter values are not natural numbers, but range over an abstract finite domain [30]. In [14], we developed a more general form of counter abstraction for expressions used in threshold guards, which leads, e.g., to the abstract domain of four values that capture the parametric intervals $[0, 1)$ and $[1, t + 1)$ and $[t + 1, n - t)$ and $[n - t, \infty)$. It is easy to see [14] that a counter abstraction simulates all counter systems for all parameter values that satisfy the resilience condition. The bound d on the diameter of counter systems implies a bound $\hat{d}$ on the diameter of the counter abstraction. From this and simulation follows that if an abstract state is not reachable in the counter abstraction within $\hat{d}$ steps, then no concretization of this state is reachable in any of the concrete counter systems. This allows us to efficiently combine counter abstraction with *bounded model checking* [5]. Typically, bounded model checking is restricted to finding bugs that occur after a bounded number of steps of the systems. However, if one can show that within this bound every state is reachable from an initial state, bounded model checking is a complete method for verifying reachability.

[1] Indeed much research in distributed algorithms is devoted to show that certain problems are solvable only under some resilience condition, e.g., the seminal result on Byzantine fault tolerance by Pease et al. [28].