Analysis of Infinite-State Graph Transformation Systems by Cluster Abstraction*

Peter Backes and Jan Reineke

Universität des Saarlandes, Saarbrücken, Germany
{rtc,reineke}@cs.uni-saarland.de

Abstract. Analysis of distributed systems with message passing and dynamic process creation is challenging because of the unboundedness of the emerging communication topologies and hence the infinite state space. We model such systems as graph transformation systems and use abstract interpretation to compute a finite overapproximation of the set of reachable graphs. To this end, we propose cluster abstraction, which decomposes graphs into small overlapping clusters of nodes. Using `astra`, our implementation of cluster abstraction, we are for the first time able to prove several safety properties of the merge protocol. The merge protocol is a coordination mechanism for car platooning where the leader car of one platoon passes its followers to the leader car of another platoon, eventually forming one single merged platoon.

Keywords: graph transformation, abstract interpretation, parameterized verification, shape analysis, distributed message-passing systems.

1 Introduction

Distributed message-passing systems such as car platoons and drone swarms consist of an unbounded and dynamically changing number of agents. These agents act in a coordinated fashion using wireless ad-hoc networks to achieve common goals. For this purpose, they assume different roles in a logical communication topology that is established on top of the physical communication medium. These communication topologies, which consist of unidirectional channels between pairs of agents, are formed by distributed protocols that all agents execute concurrently.

The purpose of our analysis is to determine the emerging topologies, which can then be used to evaluate safety properties, ensuring that the system will never reach a state with an undesired topology.

We model such systems by graph transformation systems, i.e., graphs modified by transformation rules. Graph transformation is a lingua franca with a broad

* This work was partially supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS). See `http://www.avacs.org/` for more information.

D. D'Souza et al. (Eds.): VMCAI 2015, LNCS 8931, pp. 135–152, 2015.

range of applications in systems modeling, all of which become potential use cases for our method. Many domain-specific models can be translated automatically into graph transformation systems.

In the graph transformation framework, we represent agents as labeled nodes and communication channels and message queues as labeled, directed edges of a graph. We model the dynamics of the system, like agents sending and receiving messages, detecting each other's presence and setting up and closing communication channels, as transformation rules that are applied to the graphs. Those rules match subgraph patterns in a graph, optionally restricted by application conditions, and replace them by modified subgraphs.

The main challenges with respect to the analysis of the systems under consideration are the unboundedness of the graphs, caused by the unboundedness of the number of agents, and the concurrency of the computations of the participating agents. In particular, the state space of such systems is infinite, and naive state-space exploration cannot be used for our purpose. Instead, we use abstract interpretation, overapproximating the graphs by abstract representations of bounded size.

To compute this overapproximation, we lift rule application to the abstract level, reducing the infinite concrete state space to a finite abstract one: We match the rules on the abstract representation, partly undo the abstraction, just enough to apply the rule, and restore abstraction on the result. By fixed-point iteration, we end up with one final abstract topology, an overapproximation of all graphs the system may produce.

The crucial idea of our abstraction is to decompose graphs into overlapping, simultaneously evolving *clusters*, one per node of the graph—*cluster abstraction*. Each cluster consists of a *core node*, corresponding to the specific node under consideration, and *peripheral nodes*, corresponding to the immediate neighborhood of the core node, i.e., its adjacent nodes. We keep the edges between peripheral nodes and the core node, as well as the core node itself, completely concrete. The neighborhood of a node may be unbounded, e.g., in some protocols a leader may have an unbounded number of followers. To arrive at a finite abstract domain, we use approximated counting: two or more neighborhood nodes that are similar become one summary node in the periphery. By a three-valued abstraction, we preserve information about the neighborhood edges where possible.

We have implemented cluster abstraction in a tool called `astra`. In addition to benchmarks from the literature, ranging from red-black trees to firewalls, we successfully apply `astra` to the merge protocol. The merge protocol is a coordination mechanism for car platooning that could not be fully analyzed with previous approaches.

Outline. In Section 2, we describe the graph transformation framework our work is based upon. Section 3 introduces cluster abstraction and the computation of the corresponding abstract transformer. In Section 4 we present our tool implementation `astra` and experimental results. After discussing related work in Section 5, we conclude the paper in Section 6.