Question:

Explain in detail how the graph is an abstraction of the problem

Answer:

Infinite or very large state spaces often prohibit the successful verification of graph transformation systems. Abstract graph transformation is an approach that tackles this problem by abstracting graphs to abstract graphs of bounded size and by lifting application of productions to abstract graphs. In this work, we present a new framework of abstractions unifying and generalising existing takes on abstract graph transformation. The precision of the abstraction can be adjusted according to the properties to be verified facilitating abstraction refinement.We present a modal logic defined on graphs, which is preserved and reflected by our abstractions. Finally, we demonstrate the usability of the framework by verifying a graph transformation model of a firewall.

Introduction

Formal verification of graph transformation systems aims at statically proving or inferring properties of a graph transformation system, where such properties are typically given in some form of temporal logic. It is crucial to distinguish verification and simulation, the latter being very useful only for debugging, whereas verification establishes a property for all computations of a graph transformation system. Problems do arise when approaching this task. One such problem is the possibly infinite behaviour of a system which in most cases makes it impossible to study the whole behaviour of the system. Another problem is space: even for a finite state space, each state can be quite big to represent.They can be characterised as to which approach to graph transformation is used for modelling, which verification technique is applied, and which applications are tackled. The technique presented in feeds finite-state graph transformation systems, given as a double pushout system, to an offthe-shelf model checker to verify reactive systems. However, we face the more general problem of unbounded systems. The approaches presented in both use backwards reachability analysis for hyperedge replacement grammars trying to reach an initial graph by backwards search from a forbidden configuration.The technique is applied to mechatronic systems and ad-hoc network routing,respectively, but, unfortunately, is not guaranteed to terminate. An approximation of the behaviour of a graph transformation system in terms of Petri net unfoldings was used in to verify properties of data structures residing in the run-time heap of programs with dynamically allocated heap memory. Abstract graph transformation relies on abstract interpretation of graph transformation systems, that is, given some equivalence relation, graphs are quotiented into abstract graphs of bounded, finite size. Application of productions is then lifted to work on abstract graphs. The abstraction first introduced in summarises nodes with similar kind and number of incident edges, while the abstraction of considers similar adjacent nodes. These two abstractions are generalised in this work and put into a unifying framework. To this end, we introduce the notion of neighbourhood abstraction as a part of a general abstraction framework. For this abstraction, nodes are summarized if they have similar neighbourhood up to some radius i, parameter of the abstraction. This enables abstractions with different precisions. Additionally,the number of possible abstract graphs obtained by neighbourhood abstraction is bounded.We introduce a logic accompanying our abstractions: given a formula our abstraction guarantees that a) if the formula holds for the original graph, then it holds for the abstracted graph (preservation); and b) if the formula holds for the abstracted graph, then it holds for the original one too (reflection).