

Abstract

Graph rewriting is a formal framework describing how graphs can be transformed by applying a set of rules, called a graph rewriting system. We focus on the automated proof of termination of these systems, that is, techniques for showing that, for any graph, the application of the transformation rules of a given system cannot repeat indefinitely.

We contribute to the development of automated techniques for the termination analysis of graph rewriting systems using the double pushout (DPO) approach, by proposing four contributions.

Firstly, we extend an existing technique based on type graphs. This method assigns weights to morphisms targeting a type graph; the weight of a graph subject to transformation is defined as the sum of the weights of all morphisms from that graph to the type graph. Termination is then established by showing that the graph's weight strictly decreases with each application of a transformation rule, provided that the set of weights is well-founded. Requiring the weights to be well-founded can, however, prove limiting in practice, because the search for a suitable type graph is extremely difficult. Our extension makes this search much easier by allowing weights drawn from non-well-founded sets.

Secondly, we develop a new sufficient condition, called morphism counting. This method applies to DPO graph rewriting systems with injective rules on edge-labeled directed multigraphs. It is based on the idea that rewriting cannot last indefinitely if the number of morphisms whose domain is a given subgraph strictly decreases with each transformation.

Thirdly, we extend our morphism counting method for a more fine-grained analysis. More specifically, we consider morphisms whose domain is a specific subgraph and whose image is not included in a forbidden context.

Finally, we implement these techniques in a new software tool, *LyonParallel*. Dedicated to DPO transformation of edge-labeled directed multigraphs, this tool automatically establishes termination for examples that earlier approaches could not handle.