Available online at www.sciencedirect.com
SCIENCE DIRECT®
Electronic Notes in Theoretical Computer Science
Electronic Notes in Theoretical Computer Science 154 (2006) 1–5
www.elsevier.com/locate/entcs

# Preface

This volume contains the contributions to the Workshop on Graph Transformation for Concurrency and Verification (GT-VC 2005), held as a sstellite of CONCUR 2005, on 22 August 2005 in, San Francisco, California, USA. Here we briefly recall the aim of the workshop, and report on the discussion session.

Graph transformations are a formalism which can be used to specify—in a natural and intuitive way—systems with dynamically evolving topologies. In recent years this led to close and fruitful interactions with other areas which traditionally use text-based specification languages. These include concurrency theory as well as system verification and analysis. In both areas a current tendency of increasing the level of dynamicity and mobility in the systems which are being studied is quite apparent. For instance in concurrency theory there is an abundance of process calculi for studying mobility, reaching from the $\pi$-calculus to more recent proposals such as the ambient calculus. Static analysis—on the other hand—struggles with the verification of highly dynamic systems such as pointer structures, witnessed for instance the work on shape analysis.

In our opinion graph rewriting has something to offer to these research areas: a well-developed algebraic theory of graphs and graph transformation, as well as a variety of tools. So in order to bring together people working in this area and in order to further the cooperation with the areas of verification and concurrency, we organized the first workshop on Graph Transformation for Concurrency and Verification (GT-VC).

The workshop was co-located with CONCUR in San Francisco and took place on August 22, 2005. There were 11 submissions (4 full papers and 7 work-in-progress papers), of which 8 were accepted. Apart from the ensuing presentations, Carolyn Talcott (SRI International) presented some of her work on "Formal Modeling via Executable Specifications in Rewriting Logic". The


doi:10.1016/j.entcs.2005.03.028

workshop was closed by a discussion session, reported below, in which the participants related their respective research areas to each other and placed their research in a broader context.

We are very glad for having the opportunity of disseminating the results of the workshop through ENTCS. At this opportunity we would also like to thank the organiser of the main event, Luca Alfaro, for his tireless support, and the Program Committee members (and the secondary reviewers) for their careful work:

- Paolo Baldan, University of Venice
- Ahmed Bouajjani, University of Paris 7
- Hartmut Ehrig, Technical University of Berlin
- Wan Fokkink, Vrije Universiteit Amsterdam
- Dirk Janssens, University of Antwerp
- Ian Mackie, King's College, London
- Ugo Montanari, University of Pisa
- Mooly Sagiv, Tel-Aviv University
- Vladimiro Sassone, University of Sussex
- Dániel Varró, Budapest University of Technology and Economics
- Nobuko Yoshida, Imperial College, London

## Discussion Session

The discussion was held under the motto: "Graph Transformation for Concurrency and Verification: What Gains?" where the theme of the workshop was put into a broader context. The participants were asked to position themselves with respect to the areas of graph transformation (on the one hand) and verification and concurrency (on the other hand); moreover, they were asked to state their longer-term ambitions.

### *From graph transformation to verification and concurrency or back?*

With respect to the first subject: the title and theme of the workshop — Graph Transformation for Verification and Concurrency — joins two long-standing fields of research; however, the "for" in the title can be read either from left to right — meaning that the primary aim of the research is to bring results and insights from graph transformation (abbreviated by GT) to bear on verification or concurrency theory (abbreviated by VC) — or in the reverse direction. The participants were asked which of these directions they associate with most.

It turned out that there were adherents of both directions, but also that some of those present felt that the distinction did not do justice to their actual position.

### GT → VC.

Those more interested in bringing techniques from graph transformation to verficiation and concurrency typically start with the observation that graphs are a very suitable candidate for an abstract model of computation which (due to the built-in consequence of working up to graph isomorphism) automatically smoothes over distinctions that complicate the theory otherwise. An example of this is structural equivalence of process algebraic terms, which is reduced to isomorphism of graphs in a suitable translation.

Of the contributions in the workshop, [1,4,7] most clearly represent this line of research. In each of these papers, graph-like models are used to provide semantics to process calculi.

### VC → GT.

Those more interested in bringing techniques from concurrency and verification to graph transformation, on the other hand, typically start with the observation that graphs are a very suitable model for concrete systems. The fundamental question of how to verify such systems is, however, not really affected by the choice of using graphs for modelling them. Thus, the theories and techniques developed originally in the context of verification and concurrency are to be transferred to the graph world. Typical examples of such techniques are model checking, compositionality and performance analysis.

Of the contributions to the workshop, [6,8] most clearly represent this line of research: the first is carried out in the contet of model checking, whereas the second is concerned with performance analysis.

### GT ∪ VC → SE.

It also turned out that for several of the participants, the application of their research in software engineering (abbreviated by SE) and programming is the central driving force, and the techniques brought to bear, whether from graph transformation or verification or concurrency, are of secondary importance. For instance looking at the contributions to the workshop, [2,10] propose graph transformation-based formalisms as models for the specification of algorithms, and [3] addresses system verification on the level of programs.

*Our 10-Year Ambition*

The longer-term ambition that we asked for elicited two types of answers from the participants: application-centric and theory-centric ones.

The main inspiration for the theory-centric position was the desire to find natural abstractions that will give rise to a deeper understanding of the fundamentals of computation. It was thought to be difficult to present the essential ideas of graph transformation to a non-expert as long as we are still lacking pieces of the theoretical puzzle. For instance, we know that there are many variations and distinctions in the current definitions of grphs and graph transformation; how should we give a uniform and understandable exposition to a software engineer, in a way that will enable him to make use of graph transformation in practical applications, as long as we have not achieved a more uniform theoretical understanding?

Those harbouring more application-centric ambitions tended to observe that graphical formalisms are a natural, intuitive and (nowadays) widely accepted formalism used in software engineering. Here lies a chance for graph transformation to actually make an impact. The fact that there is no consensus on what is the “best” definition of graphs or of graph transformations should not keep us from spending effort to show the usefulness of the basic ideas for practical applications.

*Arend Rensink*
*Reiko Heckel*
*Barbara König*

## References

[1] M. Bundgaard and T. Hildebrandt. Bigraphical semantics of higher-order mobile embedded resources with local names. This volume.

[2] B. Derbel and M. Mosbah. Distributed graph traversals by relabelling systems with applications. This volume.

[3] M. Dodds and D. Plump. Extending C for checking shape safety. This volume.

[4] F. Gadducci and A. Lluch-Lafuente. Graphical verification of a spatial logic for the $\pi$-calculus. This volume.

[5] R. Heckel, B. König, and A. Rensink, editors. *Graph Transformation for Verification and Concurrency (GT-VC)*, number 05–34 in CTIT Technical Reports. Centre for Telematics and Information Technology, University of Twente, 2005.

[6] H. Kastenberg. Towards attributed graphs in GROOVE — work in progress. This volume.

[7] I. Lanese and U. Montanari. Hoare vs Milner: comparing synchronizations in a graphical framework with mobility. This volume.

[8] S. Menge and G. Lajios. A framework for stochastic system modelling and analysis — work in progress. This volume.

[9] C. Talcott. Formal modeling via executable specifications in rewriting logic. In Heckel et al. [5], page 1.

[10] D. Wroblewski. Semi-local model of computations on graphs to break the local symmetry — work in progress. This volume.