



COMPOSITIONAL VERIFICATION OF
COMPONENT-BASED HETEROGENEOUS SYSTEMS

Yan Jin

A THESIS SUBMITTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN THE SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF ADELAIDE

January 2004

Abstract

It is widely acknowledged that mathematically-based formal specification and verification methods can provide substantial help in revealing ambiguity and inconsistency of informal system descriptions, and increase confidence in the correctness of system designs. However, as their complexity rapidly grows, modern computer-based systems often consist of parts (or subsystems) with very different characteristics, *e.g.* data-centred vs. control-oriented. Traditional formal methods relying on a single specification language are no longer appropriate for coping with all aspects of such a system. A combination of languages to be used for system specification is required.

To meet this need, it is essential to devise sound principles and techniques for integrating various languages and for reasoning about the resultant component-based heterogeneous system. It is also essential to provide mechanical support for their formal verification. This thesis focuses on discrete-event systems and a class of specification languages, *viz.* graph-like visual languages. It takes a two-step approach to study these fundamental issues and seek practical, tool-supported solutions.

Firstly, an underlying methodological framework is proposed, which enables both the use of different languages (or formalisms) to describe different subsystems and the automatic verification of heterogeneous systems. This framework is built on a solid semantic base of interconnected discrete-event components. It uses a semantic interpretation approach to heterogeneous systems, specifying language-specific interpreters which enable heterogeneous components to be defined in terms of this semantic base. More specifically, the interpreters are parameterized with component models and execute on behalf of the components according to the semantics of the specification languages. They are also enhanced with facilities that allow analysis tools to access the state and transition information of the components and control their concurrent execution. As a

consequence, not only is the exhaustive analysis of heterogeneous systems supported, but also an open and extensible platform is provided, which allows various graph-like languages and formal verification techniques to be used for specifying and reasoning about heterogeneous systems.

Secondly, this thesis focuses on model checking, a robust and largely automatic approach to system formal verification. It proposes a compositional approach to combat the state space explosion problem, a well-known obstacle to model checking. This approach divides a verification problem of a system into sub-problems of its components and then addresses each sub-problem independently. The key is to specify abstract communication protocols for components using a lightweight formal language — *interface automata* [54], and then utilise these protocols as behavioural contracts for independent analysis of the components and their composition patterns. It is demonstrated that, adopting this divide-and-conquer approach, not only the basic properties of component-based systems, such as consistency and deadlock freedom, but also their safety properties can be proved. Furthermore, this compositional approach is implemented as automated tools in the context of the Moses tool suite [65], utilising the semantic interpretation approach proposed earlier. These tools are then applied to the verification of a non-trivial distributed embedded system — the Production Cell [138]. It is shown that a significant reduction in complexity of system verification is achieved.

Contents

Abstract	i
Declaration	iii
Acknowledgements	iv
1 Introduction	1
1.1 Motivation	1
1.2 Research Goal	3
1.3 Contributions	4
1.4 Thesis Outline	6
1.5 Relation to Previous Publications	7
2 Background	9
2.1 Formal Verification	9
2.1.1 Theorem Proving	10
2.1.2 Model Checking	11
2.1.3 Methods for Attacking the State Space Explosion	12
2.2 Multi-Formalism Modelling	16
2.2.1 Semantic Approaches to Heterogeneous Systems	19
2.3 Component-Based Design	22
2.4 Modelling Languages	24
3 Compositional Verification of Component-Based Systems	26
3.1 Preliminaries	27
3.1.1 Reactive Transition Systems	28

3.1.2	RTS Networks	30
3.2	Independent Analysis of Components	34
3.2.1	Discrete-Event Components	35
3.2.2	Interface Automata	37
3.2.3	Conformance of Discrete-Event Components	39
3.2.4	Practical Conformance Checking	41
3.3	Compositional Verification of DEC Networks	44
3.3.1	DEC Networks	45
3.3.2	IA Networks	49
3.3.3	Verification of Basic Properties for Closed DEC Networks	50
3.3.3.1	Consistency	50
3.3.3.2	Deadlock Freedom	52
3.3.3.3	Verifying Basic Properties	53
3.3.4	Verification of Basic Properties for Open DEC Networks	56
3.3.4.1	Consistency	56
3.3.4.2	Deadlock Freedom	59
3.3.4.3	Verifying Consistency	60
3.3.4.4	Verifying Deadlock Freedom	61
3.3.4.5	Verifying Conformance	61
3.3.5	Verifying Safety Properties for DEC networks	64
3.3.5.1	Local Properties of Components	65
3.3.5.2	System Boundedness	66
3.3.5.3	System-Wide Properties	66
3.4	Summary and Discussion	69
3.5	Related Work	70
4	Semantic Interpretation Approach to Heterogeneous Systems	74
4.1	Preliminaries	77
4.1.1	Attributed Graphs	78
4.1.2	Object Mapping Automata	78
4.1.3	Platform Assumptions	82
4.2	Behavioural Contract with Analysis Tools	83

4.3	Petri Nets	86
4.3.1	Notation	86
4.3.2	Attributed Graphs	88
4.3.3	Semantic Interpretation	89
4.4	UML Statecharts	93
4.4.1	Notation	94
4.4.2	Attributed Graphs	100
4.4.3	Semantic Interpretation	101
4.4.3.1	Static Function Declaration	102
4.4.3.2	Analysis Variable Specification	104
4.4.3.3	Initialization	105
4.4.3.4	Rule “step”	106
4.5	Summary and Related work	113
5	Implementing Analysis Tools	116
5.1	Overview of the Moses Tool Suite	116
5.1.1	Syntactic Definition	118
5.1.2	Diagram Editing	120
5.1.3	Diagram Simulation	121
5.2	Approach to Verification	121
5.2.1	State Space Exploration	122
5.2.2	Specifying Safety Properties	125
5.2.3	Verifying Safety Properties	126
5.3	Approach to Compositional Verification	127
5.3.1	Incorporating the IA Formalism	128
5.3.2	Conformance Checking for Components	131
5.3.3	Consistency Checking for IA Networks	134
5.3.4	Conformance Checking for Open Systems	135
5.3.5	Verifying System-Wide Safety Properties	137
5.4	Summary and Discussion	138

6	Case Study: the Production Cell	140
6.1	Task Description	140
6.1.1	Constituent Machines	141
6.1.2	Requirements	143
6.2	Component-Based Design	144
6.2.1	Protocol Design	145
6.2.2	Component Design	148
6.2.2.1	Elementary Motion Control	148
6.2.2.2	Feed Belt	149
6.2.2.3	Deposit Belt	151
6.2.2.4	Arms	152
6.2.2.5	Robot	154
6.3	Compositional Verification	157
6.3.1	Verifying Basic Properties	157
6.3.2	Verifying Safety Properties	158
6.3.2.1	Component Invariants	159
6.3.2.2	System Boundedness	160
6.3.2.3	System-Wide Properties	160
6.4	Summary and Related Work	163
7	Conclusions	165
7.1	Contributions	165
7.2	Future Work	168
A	The Production Cell: Further Detail	171
A.1	Remaining Component Models	171
A.1.1	Elevating Rotary Table	171
A.1.2	Press	172
A.1.3	Crane	172
A.2	Compositional Verification	176
A.2.1	Design Consistency	176
A.2.2	Restrictions of Machine Movement	177

A.2.3 Avoidance of Machine Collisions 179
A.2.4 No Blanks Dropped Outside Safe Areas 180
A.2.5 Insurance of a Sufficient Distance between Blanks 181
A.2.6 Summary of the Verification Results 182