
Abstract of PhD Thesis

Author: Alicia Villanueva
Title: Model Checking for the Concurrent Constraint Paradigm
Language: English
Supervisor: María Alpuente and Moreno Falaschi
Institute: Universidad Politécnica de Valencia, Spain *and*
Università degli Studi di Udine, Italy
Date: May 24, 2003

Abstract

Formal verification of temporal properties is necessary in many real applications. We can find in the literature many case studies which show us how formal verification techniques allowed scientists to find errors in systems that were thought to be correct. *Model checking* is an automatic formal verification technique that, given a model of a system and a temporal formula, determines whether the model satisfies the formula. This technique is based on the exhaustive analysis of the state-space of the system, therefore the main drawback of model checking software is the state explosion problem. Many approaches such as the symbolic approach, the abstract interpretation approach, the on-the-fly approach, etc. have been defined in the last years to mitigate this problem. Moreover, although model checking was defined for verifying finite-state systems, many researchers try to extend it to handle infinite-state systems.

In the dissertation described here, the Concurrent Constraint (cc) paradigm is used to specify reactive and hybrid systems. Then, we provide the necessary formalism to verify such kind of systems. In particular, we handle three of the extended languages defined from the cc paradigm: the tcc, tccp and hcc languages. The first two languages have a discrete notion of time allowing the programmer to model reactive systems. The last one introduces a continuous notion of time which allows us to model hybrid systems.

It is well known that an appropriate denotational semantics allows us to perform very interesting analysis over languages in a simple way. In that sense, in the thesis we show that, although both denotational and operational semantics were given when tcc was defined, they do not always coincide. We define a fully abstract denotational semantics (w.r.t. the operational semantics) for the tcc model. We also describe an application of our new semantics to the analysis of the expressive power of the new construct introduced in tcc to model the timeout or

preemption behaviors. We show that the new construct makes the tcc language a more powerful language than the cc model.

The main result of this thesis is the definition of a model checking algorithm for tccp programs. The idea is to exploit the good features of the cc paradigm to solve the state explosion problem of model checking. We take advantage of the *constraint* nature of the language in order to redefine the three phases of the model checking technique. First of all we use constraints to define what a state is in the model of the system. A state of the model can be seen as a conjunction of constraints. This means that a state of our model represents a set of states of a classical *Kripke Structure*.

Furthermore, constraints are directly used in the logic that we consider, thus it is not necessary to transform our model into a Kripke Structure. If we had use classical temporal logics, this transformation would be necessary since they could not handle our model directly. Note that we have only partial information while classical logics need full information about values of variables.

We also define a method to verify hcc programs. We show that the hcc language allows the programmer to specify hybrid systems in general and linear hybrid systems in particular. The key idea in this case is also to take advantage of the nature of the cc paradigm. The approach presented in this thesis is the first attempt to apply the model checking technique to the hcc language.

Table of Contents

Introduction	vii
I.1 A Model for Concurrency	viii
I.2 Formal Verification	ix
I.3 Verification and Constraints	x
I.4 The Thesis Approach	xi
I.5 The Thesis Overview	xiii
1 Preliminaries	1
1.1 Basic Set Theory	1
1.2 Domain Theory	6
1.3 Transition Systems	8
1.4 Constraint Systems	9
1.5 Closure Operators	16
2 The tcc language	19

2.1	Syntax	20
2.2	Operational Semantics	23
2.2.1	Observable Behaviours	26
2.3	Denotational Semantics	28
2.4	Applicatios	29
3	New Semantics for tcc	31
3.1	New Denotational Semantics	32
3.2	Correctness and Completeness	40
3.2.1	On the Expressive Power of tcc	47
3.3	Related Works	48
4	The tccp language	51
4.1	Syntax	52
4.2	Operational Semantics	53
4.3	Applicatios	54
4.4	tccp vs tcc	55
5	Verification Techniques	59
5.1	Theorem Proving	60
5.2	Testing	61
5.3	Model Checking	63
5.3.1	The state explosion problem	64
5.3.2	Complexity	66
5.3.3	Model checking characteristics	66
5.4	Infinite State Model Checking	67
5.5	Model Checking for the cc Paradigm	69
6	A Model for tccp programs	71
6.1	Labelling	71
6.2	The tccp Structure	72
6.3	Construction of the model	74
6.4	Correctness and Completeness	81

6.5 The tcc Approach	83
7 The Specification of the Property	85
7.1 Temporal Logics	85
7.2 A Logic over Constraints	87
7.2.1 Some Examples	89
8 The Model Checking Algorithm	91
8.1 The Closure of the Formula	92
8.2 The Model Checking Graph	94
8.3 The Searching Algorithm	96
8.4 Complexity and Related Works	98
9 Hybrid cc language	101
9.1 Syntax	101
9.2 Operational Semantics	103
9.3 Applicatios	104
10 Hybrid cc Model Checking	107
10.1 Modeling	107
10.1.1 Labelling	109
10.2 Graph Construction	109
10.3 Transformation	115
11 Conclusions	119
Bibliography	123
Index	135

Author's correspondence address Alicia Villanueva
DSIC, Universidad Polit3cnica de Valencia
Camino de vera, s/n
E-46022 - Valencia
Spain