Theoretical Computer Science 523 (2014) 1–36

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Verification of gap-order constraint abstractions of counter systems [☆]

Laura Bozzelli [a,*], Sophie Pinchinat [b]

[a] Technical University of Madrid (UPM), 28660 Boadilla del Monte, Madrid, Spain
[b] IRISA, Campus de Beaulieu, 35042 Rennes Cedex, France

A R T I C L E   I N F O

Article history:
Received 7 November 2012
Received in revised form 20 November 2013
Accepted 2 December 2013
Communicated by P. Aziz Abdulla

Keywords:
Abstractions of counter systems
Temporal logics with Presburger constraints
Model checking
Verification of infinite-state systems
Complexity and decidability issues

A B S T R A C T

We investigate verification problems for *gap-order constraint systems* (GCS), an (infinitely-branching) abstract model of counter machines, in which constraints (over $\mathbb{Z}$) between the variables of the source state and the target state of a transition are *gap-order constraints* (GC) [32]. GCS extend monotonicity constraint systems [7], integral relation automata [16], and constraint automata in [19]. First, we address termination and fairness analysis of GCS. Since GCS are infinitely-branching, termination does not imply *strong termination*, i.e. the existence of an upper bound on the lengths of the runs from a given state. We show that the termination problem, the strong termination problem, and the fairness problem for GCS (the latter consisting in checking the existence of infinite runs in GCS satisfying acceptance conditions à la Büchi) are decidable and PSPACE-complete. Moreover, for each control location of the given GCS, one can build a GC representation of the set of counter variable valuations from which termination (resp., strong termination, resp., fairness) does *not* hold (resp., does *not* hold, resp., does hold).
Next, we consider a constrained branching-time logic, GCCTL*, obtained by enriching CTL* with GC, thus enabling expressive properties and subsuming the setting of [16]. We establish that, while model-checking GCS against the universal fragment of GCCTL* is undecidable, model-checking against the existential fragment, and satisfiability of both the universal and existential fragments are instead decidable and PSPACE-complete (note that the two fragments are not dual since GC are not closed under negation). Moreover, our results imply PSPACE-completeness of known verification problems that were shown to be decidable in [16] with no elementary upper bounds.

1. Introduction

Abstractions of counter systems. One standard approach in formal analysis is the abstraction-based one: the analysis is performed on an *abstraction* of the given system, specified in some weak computational formalism for which checking the properties of interest is decidable. The relation between the abstraction and the concrete system is usually specified as a semantic over-approximation. This ensures that the approach is conservative, by giving a decision procedure that (for correct systems) is sound but in general incomplete. Such a methodology has been applied in particular to the verification of counter systems which represent a widely investigated complete computational model, used for instance to model broadcast

[☆] This paper is an extended version of the results published in the proceedings of the following two conferences: (1) 13th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'12), Lect. Notes Comput. Sci., vol. 7148, Springer, 2012, pp. 88–103; and (2) 6th International Conference on Language and Automata Theory and Applications (LATA'12), Lect. Notes Comput. Sci., vol. 7183, Springer, 2012, pp. 155–168.
[*] Corresponding author.
E-mail addresses: laura.bozzelli@fi.upm.es (L. Bozzelli), Sophie.Pinchinat@irisa.fr (S. Pinchinat).

0304-3975/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.tcs.2013.12.002

protocols [23] and programs with pointer variables [11]. Counter systems extend finite-state systems by allowing a finite set of counter variables, with each counter taking values from the infinite domain of integers. Moreover, constraints on the transitions specify the relation between the variables of the target state and the variables of the source state. Though simple problems like reachability are already undecidable for 2-counter Minsky machines [28], interesting abstractions of counter systems have been studied, for which expressive classes of verification problems have been shown to be decidable. Many of these abstractions are in fact restrictions: examples include Petri nets [29], reversal-bounded counter machines [25], and flat counter systems [10,17]. Genuine abstractions are obtained by approximating counting operations by non-functional fragments of Presburger constraints between the variables of the target state and the variables of the source state. Examples include the class of Monotonicity Constraint Systems (MCS) [7] and its variants, like constraint automata in [19], and integral relation automata (IRA) [16], for which the (monotonicity) constraints (MC) are boolean combinations of inequalities of the form $u < v$ or $u \leqslant v$, where u and v range over variables or integer constants. MCS and their subclasses (namely, *size-change systems*) have found important applications for automated termination proofs of functional programs (see e.g. [7,8]). Richer classes of non-functional fragments of Presburger constraints have been investigated, e.g. difference bound constraints [18], and their extension, namely octagon relations [13], where it is shown that the transitive closure of a single constraint is Presburger definable (these results are useful for the verification of safety properties for flat counter systems). Note that difference bound constraints over (real-valued or integer-valued) variables (clocks) are also used as guards of transitions in timed automata [4]. Size-change systems extended with difference bound constraints over the natural number domain have been investigated in [6]: there, the atomic difference constraints are of the form $x - y' \geqslant c$, where c is an integer constant, and y' (resp., x) range over the variables of the target (resp., source) state. Termination for this class of systems is shown to be undecidable. To regain decidability, the authors consider a restriction, where at most one bound per target variable in each transition is allowed.

Temporal logics with Presburger constraints. An important classification of temporal logics is based on the underlying nature of time. In the *linear-time* setting, formulas are interpreted over linear sequences (corresponding to single computations of the system), and temporal operators are provided for describing the ordering of events along a single computation path. In the *branching-time* setting, formulas are instead interpreted over computation trees, which describe all the possible computations of the system from a designated initial state. Branching-time temporal logics are in general more expressive than linear-time temporal logics since they provide both temporal operators for describing properties of a path in the computation tree, and path quantifiers for describing the branching structure in computation trees.

In order to specify behavioral properties of counter systems, standard propositional linear-time temporal logics (like LTL [30]) and propositional branching-time temporal logics (like CTL* [22]) can be extended by replacing atomic propositions with Presburger constraints, which usually refer to the values of the (counter) variables at two consecutive states along a computation path (run). These enriched temporal logics allow to specify properties of counter systems that go beyond simple reachability. Hence, basic decision problems are generally undecidable. However, decidability has been established for various interesting fragments. We focus on fragments where the constraint language includes MC. For the *linear-time setting*, many decidable fragments of full Presburger LTL have been obtained either by restricting the underlying constraint language, see e.g. [19,21], or by restricting the logical language, see e.g. [12,17]. In particular, satisfiability and model checking (w.r.t. constraint automata) of standard LTL extended with MC are decidable and PSPACE-complete [19] (which matches the complexity of LTL). For the *branching-time setting*, to the best of our knowledge, very few decidability results are known. The extension of standard CTL* with MC, here denoted by MCCTL*, has been introduced in [16], where it is shown that model checking IRA against its existential and universal fragments is decidable (by contrast, model checking for the full logic MCCTL* is undecidable, even for its CTL-like fragment[1]). As done in [21], adding periodicity constraints and the ability for a fixed $k \geqslant 1$, to compare the variable values at states of a run at distance at most k, decidability of the above problems is preserved [14]. However, no elementary upper bounds for these problems are known [16,14]. Moreover, it is shown in [20] that model checking a subclass of *finitely-branching flat* counter machines w.r.t. full Presburger CTL* is decidable. In this subclass of systems, counting acceleration over every cycle in the control graph is Presburger definable. Thus, since the relation between the variables at the current and next state is functional and the control graph is flat (i.e., it contains only simple cycles), Presburger definability can be extended in a natural way to the set of states satisfying a given formula.

Our contribution. We investigate verification problems for an (infinitely-branching) abstract model of counter machines, we call *gap-order constraint systems* (GCS), in which constraints (over $\mathbb{Z}$) between the variables of the source state and the target state of a transition are (transitional) *gap-order constraints* (GC) [32]. These constraints are positive boolean combinations of inequalities of the form $u - v \geqslant k$, where u, v range over variables and integer constants, and k is a natural number. Thus, GC can express simple relations on variables such as lower and upper bounds on the values of individual variables, and equality, and gaps (minimal differences) between values of pairs of variables. GC have been introduced in the field of constraint query languages (constraint Datalog) for deductive databases [32], and also have found applications in the analysis of safety properties for parameterized systems [1,2], and for determining state invariants in counter systems [24]. As pointed

[1] Quantification over variables can be simulated by the path quantifiers of the logic.