PhD subject:
Abstract Interpretation for Logical Clocks

Frédéric MALLET and Marie PELLEAU

1 Context

The *concrete semantics* of programs formalizes the set of all possible executions of a program in all possible environments. In most cases, this set of possible executions is infinite and is not computable. *Abstract interpretation* consists in considering an abstract semantics, that is a superset of the concrete program semantics [4]. A invariant is a property that holds on all the executions and an abstract domain is a computer representation of a given category of invariants.

Our work focuses on the definition of sound models to describe the temporal behavior of programs. We rely on a set of mostly independent logical clocks [7] to characterize the valid and invalid behaviors of reactive systems [6]. Clocks evolve independently of each other, and not necessarily at regular rates, unless otherwise constrained. We use the Clock Constraint Specification Language (CCSL [9]) to specify both synchronous and asynchronous constraints and build so-called polychronous specifications [8].

Each CCSL constraint defines a set, usually infinite, of possible behaviors. The expected global behavior of the reactive system is the intersection of all sets of behaviors of all the constraints. The problem of scheduling a CCSL specification, is to, given a finite set of constraints, tell whether or not the intersection of all behaviors of all constraints is empty. If not empty, then the challenge is to represent at least some, possibly all, the valid behaviors so that their properties can be further studied. While it is acceptable to get only an under-approximation of valid schedules, we cannot afford to produce an over-approximation. As such, CCSL constraints can be considered as temporal invariants over the system behavior, and families of constraints may be characterized by abstract domains.

Abstract Interpretation has been studied in the context of temporal calculi and logics [5] and in the particular context of embedded critical software [3]. However, a purely linear or even branching-time model of time à-la-CTL is not entirely satisfactory to capture the polychronous nature of logical clocks and clock constraints seem to describe different kinds of abstract domains.

2 Goal of the thesis

The goal of the thesis is study the use of abstract interpretation in the context of logical clock specifications. While CCSL should give a first minimal setting for clock constraints, the overall scope may go beyond CCSL and extend to more general synchronous and asynchronous constraints such as found in synchronous languages [1]. The objective is to get both an adequate abstraction to get a simple-enough computer representation of polychronous specifications, but also to get, as a side effect, possibly efficient analysis verification tools to schedule multi-clock specifications [2].

In particular, we would like to customize a generic constraint solver based on abstract domains [10] and compare to state-of-the-art solutions relying on industrial SMT solvers [14] or ad-hoc solvers[11]. It should bring a more general framework to analyse logical specifications and pinpoint some particularly interesting behaviours like periodic schedules [13] or bad paths [12].

References

[1] Albert Benveniste, Paul Caspi, Stephen A. Edwards, Nicolas Halbwachs, Paul Le Guernic, and Robert de Simone. The synchronous languages 12 years later. *Proceedings of the IEEE*, 91(1):64–83, 2003.

[2] Gérard Berry and Ellen Sentovich. Multiclock esterel. In Tiziana Margaria and Thomas F. Melham, editors, *Correct Hardware Design and Verification Methods, CHARME*, volume 2144 of *Lecture Notes in Computer Science*, pages 110–125. Springer, September 2001.

[3] Julien Bertrane, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival. Static analysis by abstract interpretation of embedded critical software. *ACM SIGSOFT Software Engineering Notes*, 36(1):1–8, 2011.

[4] P. Cousot. Semantic foundations of program analysis. In S.S. Muchnick and N.D. Jones, editors, *Program Flow Analysis: Theory and Applications*, chapter 10, pages 303–342. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1981.

[5] P. Cousot and R. Cousot. Temporal abstract interpretation. In *27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 12–25, Boston, Mass., January 2000. ACM Press, New York, NY.

[6] Nicolas Halbwachs. Synchronous programming of reactive systems. In Alan J. Hu and Moshe Y. Vardi, editors, *Computer Aided Verification, CAV'98*, volume 1427 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 1998.

[7] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558—-565, July 1978.

[8] Paul Le Guernic, Jean-Pierre Talpin, and Jean-Christophe Le Lann. POLYCHRONY for system design. *Journal of Circuits, Systems, and Computers*, 12(3):261–304, 2003.

[9] Frédéric Mallet, Charles André, and Robert de Simone. CCSL: specifying clock constraints with UML/Marte. *Innovations in Systems and Software Engineering*, 4(3):309–314, 2008.

[10] Marie Pelleau, Antoine Miné, Charlotte Truchet, and Frédéric Benhamou. A constraint solver based on abstract domains. In Roberto Giacobazzi, Josh Berdine, and Isabella Mastroeni, editors, *Verification, Model Checking, and Abstract Interpretation, VMCAI*, volume 7737 of *Lecture Notes in Computer Science*, pages 434–454. Springer, January 2013.

[11] Judith Peters, Robert Wille, Nils Przigoda, Ulrich Kühne, and Rolf Drechsler. A generic representation of CCSL time constraints for UML/MARTE models. In *52nd Annual Design Automation Conference, DAC*, pages 122:1–122:6. ACM, June 2015.

[12] Ling Yin, Jing Liu, Zuohua Ding, Frédéric Mallet, and Robert de Simone. Schedulability analysis with CCSL specifications. In Pornsiri Muenchaisri and Gregg Rothermel, editors, *20th Asia-Pacific Software Engineering Conference, APSEC*, pages 414–421. IEEE Computer Society, December 2013.

[13] Min Zhang, Feng Dai, and Frédéric Mallet. Periodic scheduling for MARTE/CCSL: theory and practice. *Sci. Comput. Program.*, 154:42–60, 2018.

[14] Min Zhang, Fu Song, Frédéric Mallet, and Xiaohong Chen. Smt-based bounded schedulability analysis of the clock constraint specification language. In Reiner Hähnle and Wil M. P. van der Aalst, editors, *Fundamental Approaches to Software Engineering, FASE/ETAPS*, volume 11424 of *Lecture Notes in Computer Science*, pages 61–78. Springer, April 2019.