Project 22703: SYRF

Deliverable 7.3

Hybrid Systems

---

1 Introduction

This work package deals with analysis of properties arising as a result of interaction between a synchronous program and its physical environment. Thus, hybrid systems, mathematical models with both discrete and=20 continuous components, have been a subject of study. During the first two years of the project there were activities in two=20 directions:

- definition of formalisms and methods supporting the development process of such systems
- application to a case study provided by Saab Aerospace, where=20 temperature and flow in a climatic chamber was subject to regulation and monitoring

In the third year of the project we have continued the application related experiments using one of the abstractions referred to in the deliverables for the second year: the mode-automata model of the closed loop system including the plugged in discrete-time models of the environment in each discrete mode. These are discrete abstractions of the hybrid models developed methodically. An interesting question which arises is: how the choice of sampling rate (known to be important for= =20 modelling and analysis in the control context) affects even the modelling and analysability in the formal verification context.

In this work we experimented with the newly developed tool Lucifer and the verifcations were performed using the=20 formal verification tool NP-Tools (see Annex 7.3.A).

In addition, some efforts were spent on the question of analysis for hybrid models, though in a different direction than that originally include= d in the work plan.=20

2 Deviation from work plan

In the project description we had proposed the use of=20 linear hybrid automata as one abstraction in year two and the connection to= =20 the POLKA tool for symbolic model checking of such models. This track was not pursued due to two reasons.

First, the case study deriven transformations from a general model of hybrid systems to HA was shown to be inconsistent with modular development of environment models. Deriving the HA plant model required restrictions in the controller model which were not present in the code given for the mentioned case study (see deliverable 7.2).=20 Secondly, the work on the verification work packages was not focused on the POLKA tool in the first two years, making it less suitable as verification= =20 environment in year 3.

Instead, we developed an analysis environment whereby discrete models in Signal and continuous models in Matlab-Simulink can be co-simulated on t= he level of automatically generated C-code (see Annex 7.3.B). This co-simulation environment was then shown to be useful for illustration of undesired properties in a more complex plant than that present in the Saab Case study: a Siphon pump machine with non-linear dynamics and discret= e mode changes both in the plant and in the controller.

The rest of this report deals with the above two areas of work in more detail.

3 From continuous models to Lustre (Signal)

This approach is based on the transformation scheme depicted in figure 1, prior to verification. Of course due to the similarity in styles for Lustre and Signal, this approach is also applicable to Signal. However, the translation schemes and tools used were concretely based on Lustre in this project.=20

First, the plant model is "simplified" by restricting *some* inputs to piece-wise constant signals – replacing the non-linear continuous dynamics with discrete modes and linear dynamics in each mode. =20 Then, using the knowledge that the remaining input signals are control signals issued by a (sampled) synchronous controller,=20 each linear differential equation in each mode is transformed=20 to a difference equation (Lustre programs with real variables).

Finally, using the same scheme applied for compilation of mode-automata to flat Lustre (see workpackage 2), we=20 obtain a multi-mode Lustre model of the plant. This model is now equivalent to the original model modulo restrictions on (physical) input signals.=20

To do the analysis on the closed loop system, specially with regard to bounded response, we need to get a model of both plant and controller in a verification tool. In year three of the project we have experimented= =20 with the translator Lucifer in order to obtain a logical representation of= =20 the transition function for the controller. Annex 7.3.A provides some ideas= =20 for what can be improved in the translator prototype based on=20 this attempt.

4 Signal-Simulink Co-simulation

This approach aims to improve the current engineering practice by combining two existing techniques currently used in isolation.=20 It is based on SIGNAL and SIMULINK to model complex=20 hybrid systems. The choices are motivated by the fact that SIGNAL$\bar{2}$0 is a powerful tool for modelling complex discrete behaviours and SIMULINK is well-suited to deal with continuous dynamics.

The synchronous family of languages and associated tools provide support in programming the controller in many ways. To begin with, they provide an architectural view of the program in terms of hierarchical state machines or block diagrams.=20 In recent years, certain modelling environments for continuous systems=20 have also been augmented with versions inspired by these languages, e.g. MATLAB STATEFLOW [6] and MATRIXX [4]=20 discrete-time superblocks.

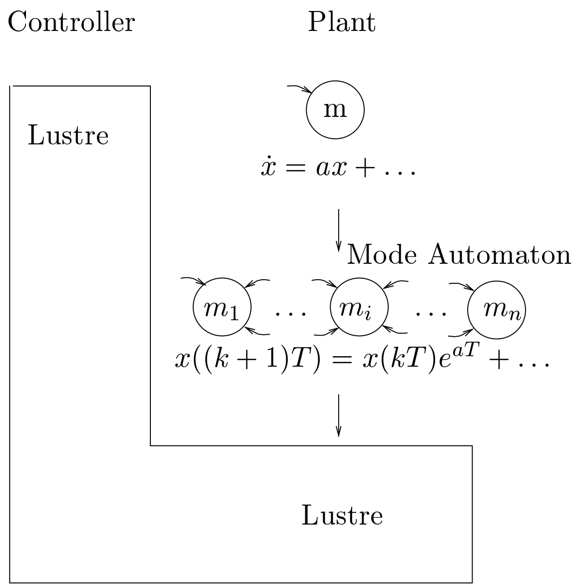

Figure 1: A transformation scheme for obtaining flat Lustre programs for a closed loop system, consisting of a sampled controller and a mode-automaton for a plant under some restrictions.

In addition, due to formal semantics, synchronous languages are used for formal analysis of the controller. However, the intermediate code and mechanisms involved go beyond this formal analysis. The results of such analyses are=20 used at later stages of development – in particular, for automatic code generation (code optimisation) and code=20 distribution [1, 2, 3, 5]. Note that these types of formal analysis are so far not supported=20 in the continuous modelling environments mentioned above.

In the third year of the project we have studied combination of these=20 benefits with a typical continuous analysis tool (the MATLAB simulation environment SIMULINK). We have studied various alternatives for implementing the communication between the plant and the controller, and how the MATLAB code generation mechanism in Real-time Workshop can be used for this purpose.

An environment for co-simulation of the plant and the controller and a=20 documentation of this study as an 80 page report is the result of this work [7]. Annex 7.3.B of this work package gives an introduction to the design of the co-simulation environment and the application of the work to a non-trivial siphon pump machine in which both the plant and the controller exhibit discrete mode changes. Also, the plant has=20 a non-trivial model and the requirements are challenging for the state of the art in formal verification of hybrid systems.

References

[1] A. Benveniste, B. Caillaud, and P. Le Guernic. Compositionality in Dataflow Synchronous Languages: Specificatio= n and Distributed Code Generation. *Information*

and Computation. To appear.

[2] A. Benveniste, B. Caillaud, and P. Le Guernic. From Synchrony to Asynchrony. In J.C.M. Baeten and S. Mauw, editors, *Proceedings of the 1= 0th International Conference on Concurrency Theory, CONCUR'99, LNCS 1664*, p= ages 162–177. Springer Verlag, 1999.

[3] T. Gautier and P. Le Guernic. Code generation in the SACRES project. In F. Redmill and T. Andersson, editors, *Towards System Safe= ty, Proceedings of the Safety-critical Systems Symposium, SSS'99*, pages 127–149, Huntingdon, UK, February 1999. Springer Verlag.

[4] Integrated Systems Inc. *SystemBuild v 5.0 User's Guide*. Santa Clara, CA, USA, 1997.

[5] A. Kountouris and C. Wolinski. Hierarchical conditional dependency graphs for mutual exclusivene= ss identification. In *12th International Conference on VLSI Design*, Goa, India= , January 1999.

[6] The MathWorks, Inc. *Stateflow User's Guide*, May 1997.

[7] S. Tudoret. Signal-simulink: Hybrid system co-simulation. Technical Report http://www.ep.liu.se/ea/cis/1999/020, Link=F6pin= gs University E-press, 1999. To appear as electronic publication.