FROM HYBRID SYSTEM SIMULATION TO REAL-TIME IMPLEMENTATION

Rachid Djenidi, Christophe Lavarenne, Ramine Nikoukhah, Yves Sorel and Serge Steer
INRIA Rocquencourt BP105 - 78153 Le Chesnay Cedex, France
Email: ramine.nikoukhah@inria.fr

11th European Simulation Symposium and Exhibition, Erlangen-Nuremberg, oct 1999

KEYWORDS

Dynamic modelling, Real-time, Multiprocessors, Hybrid simulation.

ABSTRACT

Scicos is a hybrid system simulator. In the context of automatic control, such systems are usually obtained by interconnecting a model of the environment with the model of the controller. Simulation is used to validate the controller which is usually designed using a simplified model of the environment. But even if the controller does perform properly during simulation, there is no guarantee that it can actually be used. For that, the controller algorithm must be implemented on the available hardware and satisfy the real-time constraints. SynDEx is a specialized software used for optimized real-time implementation on multiprocessor architectures. Scicos-SynDEx interface provides an environment in which controller implementability can be easily tested and controller parameters adjusted iteratively, if necessary; not to mention the generation of the real-time code for the controller hardware. This interface not only speeds up the design process, but also assures that the real-time code has the same properties as the controller model used for the simulation.

INTRODUCTION

In this paper, we present a software environment based on the hybrid system simulator Scicos and the optimized real-time code generation software SynDEx. In this environment, controllers can be designed, tested in a realistic environmental model, and real-time code for the implementation of the controller can be generated on the desired hardware (target architecture), which is often distributed.

SCICOS

Scicos (Scilab Connected Object Simulator) is a Scilab package[1] for modeling and simulation of dynamical systems including both continuous and discrete time subsystems. Scicos includes a graphical editor for constructing models by interconnecting blocks (representing predefined basic functions or user defined functions) (Nikoukhah and Steer 1997).

Scicos blocks can have input and output ports of two types: regular and activation. Input ports receive Scicos signals and output ports generate them. Scicos blocks are activated by activation signals which are received on their activation input ports. A block with no input activation port is permanently active (called time dependent) otherwise it inherits its activation times from the union of activations times of its regular input signals.

Associated with each signal, in Scicos, is a set of time intervals, called activation times, on which the signal can evolve (Benveniste 1998). Outside their activation times, Scicos signals remain constant (see Figure 1). The activation time set is a union of time intervals and isolated points called events.

Signals in Scicos are generated by blocks activated by activation signals. An activation signal causes the block to evaluate its outputs and new internal states as a function of its inputs and previous internal states. The output signals inherit their activation time sets from the generating block.

The outgoing signals of output activation ports are activation signals generated by the block. Consider for example, the Clock block which generates an activation signal composed of a train of regularly spaced events in time. If the output of this block is connected to the input activation port of a scope block, it specifies at what times the value of the inputs of the scope must be sampled for display.

[1]Free Matlab like software available by anonymous ftp from *ftp.inria.fr* in directory *INRIA/Scilab*.

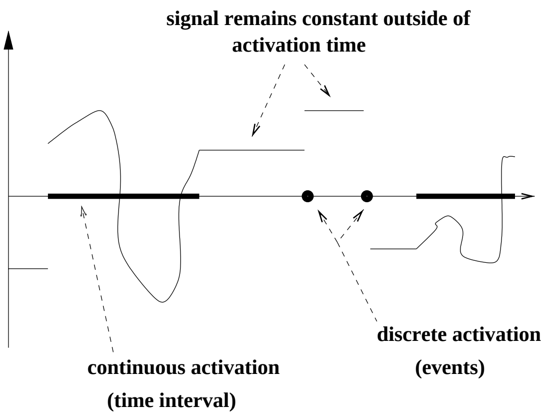
signal remains constant outside of activation time
continuous activation (time interval)
discrete activation (events)

Figure 1: A Scicos signal and its activation time set.

SYNDEX

SynDEx[2] is a system level CAD software for the implementation of distributed real-time applications. It supports the AAA methodology (Sorel 1994). AAA means Algorithm Architecture ”Adequation”[3]. The goal of the AAA methodology is to find the best matching between an algorithm and an architecture, while satisfying real-time constraints.

This methodology is based on graph models to exhibit both the potential parallelism of the algorithm, and the available parallelism of the multiprocessor architecture. The implementation consists in distributing and scheduling the algorithm data-flow graph on the multiprocessor hyper-graph while satisfying real-time constraints. This is formalized in terms of graphs transformations. Heuristics are used to optimize the real-time performances and resources allocation of embedded real-time applications.

The result of graphs transformations is an optimized Synchronized Distributed Executive (SynDEx), automatically built from a library of architecture dependent executive primitives composing the executive kernel (Grandpierre et al. 1999). These primitives support boot-loading, memory allocation, interprocessor communications, sequentialization of user supplied computation functions and of interprocessor communications, and inter-sequences synchronizations. There is one executive kernel for each supported processor: SHARC - ADSP 21060, TMS 320C40, Transputer - T80X, i80386, i8051 , i80C96, MC 68332, and UNIX workstations for now. Executive kernels for other processors can be easily ported from the existing ones.

The distributing and scheduling heuristics as well as the predicted real-time diagram, help the user to parallelize his algorithm and size the hardware while satisfying real-time constraints. Moreover, as the executives are automatically generated with SynDEx, the user is relieved from low level system programming and from distributed debugging. This allows optimized rapid prototyping, and dramatically reduces the development cycle of distributed real-time applications.

[2]SynDEx is freely available from www-rocq.inria.fr/syndex.

[3]”Adequation” is a French word meaning an efficient matching.

COMPLEMENTARITY

The functionalities of Scicos and SynDEx are complementary. In particular, Scicos does simulation of hybrid systems and SynDEx is used to implement the controller in such a way that the real-time constraints are satisfied. For example, Figure 2 illustrates a continuous-time environment model (Plant) and the algorithm of the discrete controller. The Clock block generates the events which activate the discrete part which is the controller. It is indeed this part which can have a counter-part in SynDEx. Note that in SynDEx, the inputs and outputs of the controller are replaced respectively by sensors and activators.

The complementarity between Scicos and SynDEx can be seen by studying the notions of time, algorithm and architecture.

TIME

In Scicos, neither the environment nor the time is real, in the sense that the simulation in Scicos is not performed on the target architecture and the simulation time does not need to satisfy the real-time constraints defined by the activation time sets (or more specifically the events since only discrete blocks have counter parts in SynDEx). On the other hand, for SynDEx, the computation time should be less than or equal to the real-time defined by the Scicos events which are the real-time constraints of the problem. This is why the target architecture needs to be defined in SynDEx.

ARCHITECTURE

In Scicos, the simulation consists of the sequential execution of the controller algorithm on a workstation (or a PC), in addition to the simulation of the environment which usually requires an ordinary differential equation solver. For Scicos, the host computer on which the SynDEx program runs to optimize and generate the controller program should be distinguished from the hardware on which the code is to be implemented. It is of course this latter which is subject to real-time constraints and other constraints such as price, and in case of embedded hardware, size, weight, etc...

ALGORITHM

The controller model is a data flow graph, both in Scicos and in SynDEx, defining the controller algorithm. The function relating the inputs and the outputs of the model is of course the same in Scicos and SynDEx and the operations to be performed to realize this function and the corresponding data dependence is the same regardless of the hardware architecture used, both for simulation and for real-time execution. In Scicos, the parallelism naturally associated with data flow graphs is not used to optimize the simulation. On the other hand, SynDEx uses it to take advantage of the parallelism available in the hardware architecture to satisfy real-time constraints.

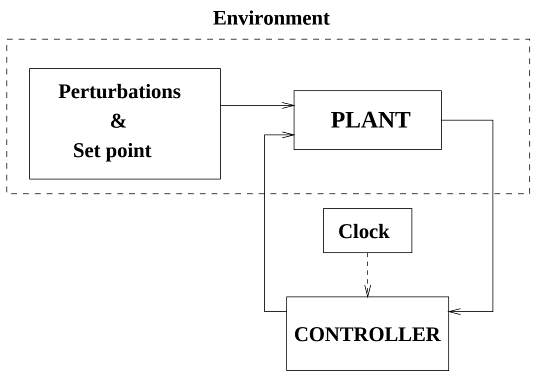

Figure 2: A typical hybrid system.

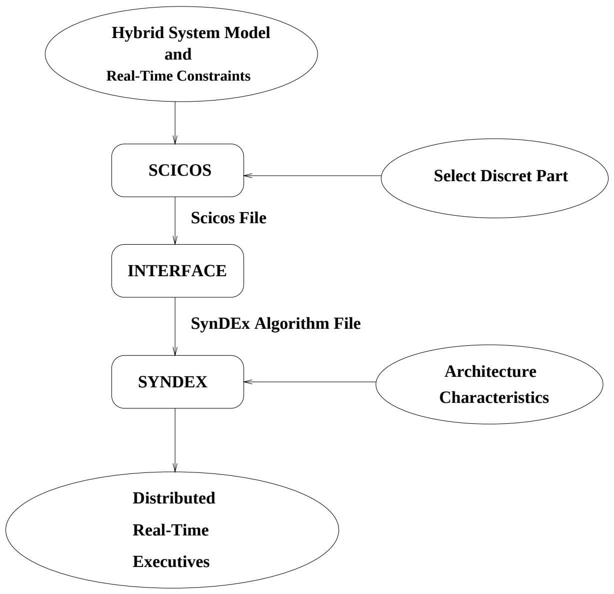

Figure 3: Unique environment

SCICOS - SYNDEX INTERFACE

As we can see on Figure 3, the input of the interface is a file generated by Scicos from the block-diagram description of the controller algorithm. The output of the interface is a file describing the same algorithm in a SynDEx input format.

Among the four types of blocks existing in Scicos, the *zero-crossing detection* and the *time-dependent blocks* concern only continuous-time behavior and thus need not and cannot be translated. For the moment, only regular blocks (*blocks executing an immediate function with possibly an internal discrete state*) are translated; translation of *Synchro blocks* (conditional blocks use for subsampling) is not yet operationnal, but will be in a near future.

Note that each Scicos block after translation can become one or more blocks in SynDEx. For example, let us consider the discrete-time linear system block Figure 4 implementing:

$$z_k = A.z_{k-1} + B.u_k \tag{1}$$

$$y_k = C.z_{k-1} + D.u_k \tag{2}$$

Here z is the state, u and y the inputs and the outputs, and A, B, C, D are given matrices.

Graph-Scicos

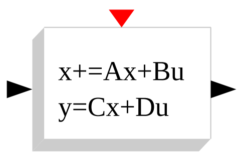

Figure 4: Scicos diagram

To express this system in SynDEx, three blocks are needed (see Figure 5). Two are associated with multiplication and addition; these blocks have for parameter the A, B, C, D matrices. These are state-less blocks realizing immediate functions. And one memory block to store the previous value of the state. The interface in this case automatically generates the three blocks and the connections yielding the SynDEx graph depicted in Figure 5.

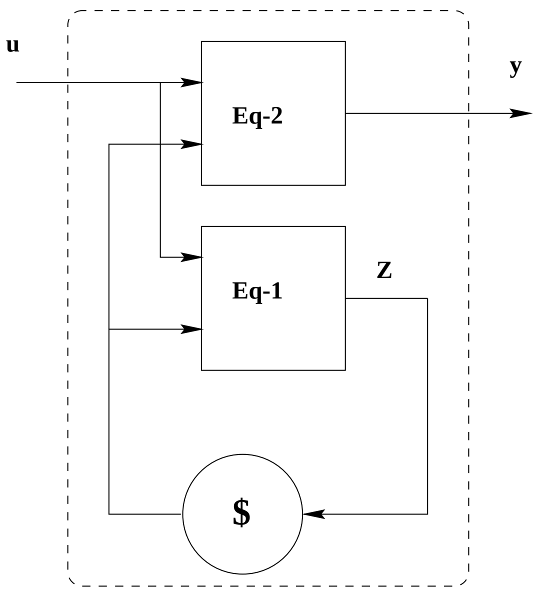

Figure 5: SynDEx graph

CONCLUSION

In this paper, we have presented a complete procedure from the design and validation by simulation of a controller algorithm to the implementation on multiprocessor architectures. In practice however this procedure does not work in a single run; controller parameters and target architectures must be ajusted iteratively to make sure the Real-Time constraints can be met. It is for this reason that the complete procedure is automated facilitating the iterative process.

REFERENCES

Benveniste, A. 1998. “Compositional and Uniform Modeling of Hybrid Systems”, *IEEE Trans. Automat. Control*, Vol. 43 (Apr.): 579-584.
Grandpierre, T.; C. Lavarenne; Y. Sorel. 1999. “Optimized Rapid Prototyping for Real-Time Embedded Heterogeneous Multiprocessors.” In *Proceedings of CODES’99* (Rome, Italy, May 3-5). ACM press, 74-78.
Nikoukhah, R.; S. Steer. 1997. “SCICOS : A Dynamic System Builder and Simulator, User’s Guide - Version 1.0”. Technical Report 0207. INRIA, (Rocquencourt, France, June).
Sorel, Y. 1994. “Massively Parallel Computing Systemes with Real Time Constraints, the Algorithm Architecture Adequation Methodology.” In *Proceedings Massively Parallel Computing Systems, the Challenges of General-Purpose and Special-Purpose Computing.* (Ischia, Italy, May).