ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 *2008. No. 1(81)*
ELEKTRONIKA IR ELEKTROTECHNIKA

AUTOMATION, ROBOTICS
T125
AUTOMATIZAVIMAS, ROBOTECHNIKA

Simulation of Mechatronic Systems using Behavioural Hybrid Process Calculus

T. Krilavičius
Department of Informatics, Vytautas Magnus University,
Vileikos str. 8, LT-44404 Kaunas; e-mail: T.Krilavicius@gmail.com

Introduction

Omnipresence of computers increased software reliability requirements considerably. Consequently, it stimulated interest in formal modelling and analysis of diverse computer systems, among which substantial position is taken by hybrid systems. Hybrid systems combine continuous evolution and instant discrete changes. Well-known examples of hybrid systems are software controlled electromechanical systems, such as microwave oven. However, pure mechanical systems, such as bouncing ball or spring-mass system can be treated as hybrid systems as well.

The growing interest in hybrid systems both in computer science and control theory has generated a new interest in models and formalisms that can be used to specify and analyse such systems. A prominent framework for hybrid systems is provided by the family of hybrid automata models (hybrid automata [1], hybrid behavioural automata [6], and hybrid input/output automata [10]). More recently, process algebraic models have been put forward as a vehicle for the study of hybrid systems [4, 2, 3].

Simulation is a *de facto* standard tool in both academia and industry for analysis of hybrid systems. It helps to detect potential weaknesses and errors, and provides information on performance of system. There is a number of simulation tools, which provide various facilities for analysis of hybrid systems. Hybrid χ [18] provides facilities for simulation of hybrid process calculus. HyVisual [9] is a Java based visual modeller and simulator for hierarchical continuous time dynamical and hybrid systems. Dymola, Stateflow/Simulink [5] and 20-Sim provide industrial strength facilities for simulation of non-causal object oriented simulation language Modelica [12], hierarchical formalism and bond graphs [17], respectively.

We report a work in progress, a technique for simulation of Behavioural Hybrid Process Calculus (BHPC) [7]. BHPC is a process calculus that extends the standard repertoire of operators that combine discrete functional behaviour with features to also represent and compose continuous-time behaviour. Dynamic behaviour is represented by the evolution of variables, which are typically defined in terms of differential equations. Following [14], behaviour can be simply seen as the set of all allowed real-time evolutions, or *trajectories*, of the system variables. The operational semantics of the calculus defines the transitions for the simulator. Choice operator provides a way to describe nondeterministic choice among processes. Concurrency is defined using parallel composition. An adapted version of the expansion law is used to solve parallel composition. Proposed simulation technique is assessed on hybrid simulator Bhave prototype, a part of Bhave toolset for hybrid systems modelling, analysis and simulation.

Behavioural Hybrid Process Calculus

In this section, we introduce main concepts of Behavioural Hybrid Process Calculus. See [7] for the details and proofs.

Trajectories

The continuous behaviour of hybrid systems can be seen as the set of continuous-time evolutions of system variables. We will call them *trajectories*. We assume that trajectories are defined over bounded time intervals $(0,t]$, and map to a *signal space* to define the evolution of the system. The signal space **W** specifies the potentially observable continuous behaviour of the system. Components of the signal space correspond to the different aspects of the continuous behaviour, like temperature, pressure, etc. They are associated with *trajectories qualifiers* that identify them.

Sometimes it is useful to define conditions on the *end-points* of trajectories or the *exit conditions*. We will use $\Downarrow$ to denote such conditions, as the restrictions on set of trajectories (1)

$$\Phi \Downarrow P_{\text{exit}} = \{\varphi : (0,u] \to W_1 \times \cdots \times W_n \in \Phi \mid P_{\text{exit}}(\varphi(u))\}. \quad (1)$$

Hybrid Transition System

We define a hybrid transition system as a collection $HTS = \langle S, A, \rightarrow, W, \Phi, \rightarrow_c \rangle$, where S is a *state space*. The *discrete transition relation* $\rightarrow \subseteq S \times A \times S$ defines discrete changes annotated by actions ($a \in A$). The *continuous-time* transition relation $\rightarrow_c \subseteq S \times \Phi \times S$ links continuous changes to trajectories ($\varphi \in \Phi$).

We introduce a language (2) for defining hybrid processes.

$$B ::= 0 \mid a.B \mid [f \mid \Phi] \mid \sum_{i \in I} B_i \mid B \parallel_A^H B \mid P . \qquad (2)$$

Deadlock 0 is the process that does not show any observable behaviour.

Action prefix $a.B$ defines a process that starts with action a and afterwards engages in B. *Silent actions* [11] (denoted τ) are used to specify nondeterministic behaviour.

Trajectory prefix $[f \mid \Phi]B(f)$ models behaviour of a process that executes a continuous trajectory. In trajectory prefix definition f is a trajectory variable and Φ is a set of trajectories. It takes a trajectory or a prefix of a trajectory in Φ. If a trajectory or a part of it was taken and there exists a continuation of the trajectory, then the system can continue with a trajectory from the trajectory continuations set. If a whole trajectory was taken, then the system may continue with B, too.

We will extend notation to make use of trajectory prefix more convenient $[q_1,\ldots q_m \mid \Phi \Downarrow P_{\text{exit}}]$, where $q_1,\ldots, q_m$ are trajectory qualifiers, which can be used to access corresponding parts of trajectories. Moreover, the set of trajectories can be defined in several different ways, e.g. by ODE/DAE. We will allow such notation in the trajectory prefix definition to bring out conditions on the set of trajectories. Furthermore, we will allow defining the set of trajectories directly in the definition of trajectory prefix, where commas will be used to separate conditions. We will use $\Downarrow$ to separate exit conditions, when it is required.

Choice $\sum_{i \in I} B_i$ is a generalised nondeterministic choice of processes. To generate the set we allow arbitrary index sets I. It chooses before taking an action prefix or trajectory prefix. Binary version of choice is denoted B_1+B_2.

Parallel composition $B_1 \parallel_A^H B_2$ specifies the behaviour of two parallel processes. The operator explicitly attaches the sets of synchronising action names A and of synchronising trajectory qualifiers H. Synchronisation on actions has an interleaving semantics. Trajectory prefixes can evolve in parallel only if the evolution of coinciding trajectory qualifiers is equal.

Recursion allows defining processes in terms of each other, as in the equation $B=P$, where B is the process identifier and P is a process expression that may only contain actions and signal types of B.

Only a subset of complete language is introduced in this paper. Some auxiliary functionality, such as renaming, is presented in [7].

Congruence

One of the main tools to compare systems is a *strong bisimulation*. The bisimulation for continuous dynamical systems is presented in [19]. The process algebraic version is discussed in [11]. A strong bisimulation for hybrid transition systems requires both systems to be able to execute the same trajectories and actions and to have the same branching structure. Strong bisimulation for BHPC is a congruence relation w.r.t. all operations defined above. See [7] for the details.

Derived Operators

BHPC is an assembly language for a modelling of hybrid systems. We add auxiliary constructs to increase usability of the language.

We introduce *parameterised action prefix* $a_v.B = \sum_{v \in V} a_v.B(v)$ for convenience (as in [11]), e.g., for value passing.

Sometimes it is useful to check some conditions explicitly, and if they are not satisfied, to stop the progress of process. With the *guard* construct $\langle Pred \rangle.B$, these conditions can be given as a predicate.

Idling in BHPC is defined as $\text{idle} = [t \mid \dot{t} = 0]$, where t is a reserved variable. It does not manifest any observable behaviour, but reacts as soon, as another process, which communicates with the process, which follows the idling period, invokes the follow-up process.

Simulation of BHPC

Simulation of hybrid systems combines continuous and discrete simulation. Consequently, it should include elements from both areas.

Simulation of discrete systems is usually substituted by testing or verification in computer science. In contrast, continuous simulation is regularly used in industry and is well established scientific research object.

Unfortunately, it is not possible just to put together discrete and continuous simulation techniques to get hybrid simulation, because, in addition, it is necessary to incorporate interaction of discrete and continuous worlds.

Our proposed technique for simulation of BHPC is based on traditional process algebraic simulation, i.e. on the correct application of *structural operation semantics* (SOS) rules [13]. Essentially, operational semantics provide a definition of stepwise execution of process algebraic expression. Rules define all allowed executions for each operator. However, some expressions require recursive transformation before it is possible to simulate a step. In addition, parallel composition requires a special treatment. Usually an expansion law is used to resolve it (we also use it), but in some cases *linearization* [16] is used.

We present an abstract description due to space limitations, see [7] for the proofs and details.

1. The system is initialised.
2. While a current simulation time is less than the maximal simulation time and no other stop conditions

have occurred, a transition is taken and the state is updated.

a. The process is transformed to so-called *normal form*.

b. A non-deterministic choice is made to proceed with discrete or continuous transition.

i. If the discrete transition is chosen, then one of the discrete transitions is nondeterministically selected and simulated.

ii. If the continuous transition is chosen, then one of the continuous transitions is simulated.

3. When the simulation time has finished or other stop condition has occurred, it is halted.

Discrete transition part is rather trivial. Taking continuous transition involves certain complex symbolical transformations of equations, and is limited by (external) ODE/DAE solver, but is rather standard in continuous simulation. Procedure of expression transformation to normal form is specific to process algebras and SOS rules based semantics. Most of it is almost directly based on SOS rules, and just requires thorough bookkeeping. However, parallel composition requires an expansion law that is used to express the parallel composition as a choice of processes.

Above defined algorithms are implemented in prototype tool Bhave prototype, a part of Bhave toolset. For more information about Bhave prototype and Bhave toolset (http://fmt.cs.utwente.nl/tools/bhave.), see [8, 15, 20].

Example: Simple and Controlled Thermostat

We illustrate BHPC language and simulation algorithm/tool with the following example.

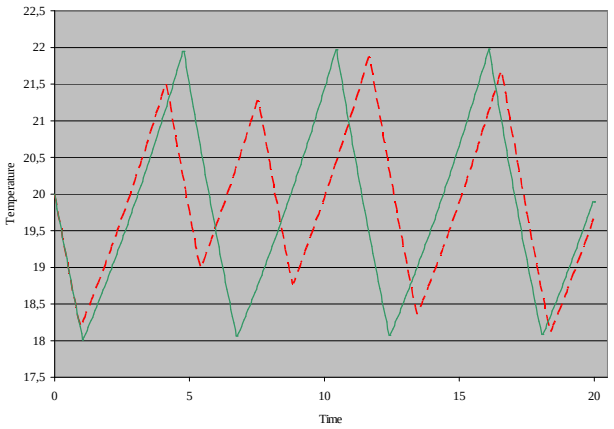

Fig. 1. Temperature changes for simple and upgraded thermostat

A simple thermostat maintains the temperature in the interval 18-22, while switching on and off in the intervals 18-19 and 21-22, correspondingly. The corresponding model in BHPC is described in (3).

In process ThOff the heater is off and the trajectory prefix defines the temperature fall. When the temperature reaches the interval [*tempOn*, *tempMin*], the process can perform action *on* and switch to the process ThOn.

Process ThOn analogously defines the period of heating.

However, it is possible to upgrade such thermostat without changing the specification itself. Let us add a controller that observes temperature and forces the thermostat to switch on and off at exactly 19 and 21, correspondingly (4).

$$
\begin{aligned}
&\mathrm{Thermostat}(l_0) = \mathrm{ThOff}(l_0);\\
&\mathrm{ThOff}(l_0) = \lfloor l \mid \Phi_{\mathrm{Off}}(l_0) \Downarrow tempOn \geq l \geq tempMin \rfloor.\\
&\qquad on.\mathrm{ThOn}(l);\\
&\mathrm{ThOn}(l_0) = \lfloor l \mid \Phi_{On}(l_0) \Downarrow tempOff \leq l \leq tempMax \rfloor.\\
&\qquad off.\mathrm{ThOff}(l);\\
&\Phi_{\mathrm{Off}}(l_0) = \{ l : (0,t] \to \Re \mid l(0) = l_0, \dot{l} = -Kl \};\\
&\Phi_{\mathrm{On}}(l_0) = \{ l : (0,t] \to \Re \mid l(0) = l_0, \dot{l} = K(h-l) \};
\end{aligned}
\tag{3}
$$

$$
\begin{aligned}
&\mathrm{Control}(l_0) = \lfloor l \mid \mathrm{any}(l_0) \Downarrow l = 19 \rfloor on,\\
&\qquad \lfloor l \mid \mathrm{any}(l_0) \Downarrow l = 21 \rfloor off.\mathrm{Control}(l),\\
&\mathrm{UpgradedThermostat}(l_0) =\\
&\qquad = \mathrm{ThOff}(l_0) \|^{l}_{on,off} \ Control(l_0),
\end{aligned}
\tag{4}
$$

where any(l) is a special function that models an observer, i.e. it accepts any behaviour for l. It works only in parallel composition. Technically it just adds exit conditions to the parallel composition of trajectory prefixes.

Results of simulation of the simple and controlled thermostat are depicted in Fig. 1. The dashed line depicts evolution of the simple thermostat and the solid line depicts evolution of the coupled version.

Conclusions

In this paper we proposed a simulation technique for Behavioural Hybrid Process Calculus. The calculus and transitions system were introduced, operators for the calculus were explained. We focussed discussion on the hybrid simulation and presentation of calculus itself, and did not delve into technical implementation details or case study.

The current tool is built not just as a prototype of an industrial tool, but as a hybrid "*sand-box*", a place to experiment with BHPC and related developments. Consequently, the architecture and implementation of the tool are being designed in such a way that it is easy to accommodate the changes in the calculus and to test the algorithms developed for hybrid systems in BHPC framework. Adaptable and well documented interfaces for ODE/DAE solvers should be provided for experimenting with different approaches for continuous-time behaviour simulation.

Our plans include further development of process algebra and simulation tool. We are planning to revise BHPC according to the tool implementation and case studies results. We would like to extend visualisation part, namely, investigate visualisation of the model itself. Especially attractive are object diagrams where objects are represented by mnemonically shaped icons. Moreover, we are investigating a novel hybrid simulation visualisation approach, message sequence plots (MSP), introduced in [8, 7]. We believe that MSP could be very useful when it is

important to see not only communication or continuous evolution separately, but to see how they influence each other.

References

1. **Alur R., Courcoubetis C., Henzinger T., Ho P.-H.** Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems // Hybrid Systems. – 1993. – Vol. 736. – P. 209–229.
2. **Bergstra J., Middelburg C.** Process algebra for hybrid systems // Theoretical Computer Science. – 2005. - Vol. 335 - P. 215-280.
3. **Brinksma E., Krilavičius T., Usenko Y. S**. Process algebraic approach to hybrid systems // Proc. of 16th IFAC World Congress. – Prague. – 2005. – P. 6.
4. **Cuijpers P., Reniers M**. Hybrid process algebra // Tech. rep. – DMCS, Tech. Univ. of Eindhoven (TU/e). – 2003. – P. 120.
5. **Hamon G., Rushby J**. An operational semantics for Stateflow // FASE. – 2004. – P. 229–243.
6. **Julius A**. On Interconnection and Equivalence of Continuous and Discrete Systems: A Behavioral Perspective / PhD thesis.– SSCG, Univ. of Twente. – 2005. – 173 p.
7. **Krilavičius T**. Hybrid Techniques for Hybrid Systems / PhD thesis.– FMT, Univ. of Twente. – 2006. – 192 p.
8. **Krilavičius T., Schonenberg M. H.** Discrete simulation of behavioural hybrid process calculus // IFM2005 Doctoral Symposium on Integrated Formal Methods. – 2005. – P. 33–38.
9. **Lee E., Zheng H**. Operational semantics of hybrid systems // Hybrid Systems: Computation and Control. – 2005. – P. 25–53.
10. **Lynch N., Segala R., Vaandrager F**. Hybrid I/O automata // Information and Computation. – 2003. – P. 105–157.
11. **Milner R**. Communication and concurrency. – Prentice-Hall, Inc. – 1989. – P. 37–43.
12. **Modelica Association**. Modelica – A Unified Object Oriented Language for Physical Systems Modelling: Language Specification. – 2005. – 137 p.
13. **Plotkin G**. A Structural Approach to Operational Semantics // Tech. Rep. DAIMI FN-19. – University of Aarhus. – 1981.– 133 p.
14. **Polderman J., Willems J. C**. Introduction to Mathematical Systems Theory: a behavioral approach. – Springer. – 1998. – 455 p.
15. **Schonenberg M. H**. Discrete simulation of behavioural hybrid process algebra / Master thesis.– Univ. of Twente. – 2006. – 125 p.
16. **Usenko Y. S**. Linearization in μCRL / PhD thesis.– Tech. Univ. of Eindhoven (TU/e). – 2002. – 188 p.
17. **van Amerongen J., P.Breedveld P**. Modelling of physical systems for the design and control of mechatronic systems // Annual Reviews in Control. – 2003. – Vol. 27, No. 1. – P. 87–117.
18. **van Beek D., Man K., Reniers M., Rooda J., Schiffelers R**. Syntax and consistent equation semantics of hybrid chi // Tech. Rep. – Tech. Univ. of Eindhoven (TU/e). – 2004.
19. **van der Schaft A**. Bisimulation of dynamical systems // HSCC. – 2004. – Vol. 2993. – P. 555–569.
20. **van Putten A**. Behavioural hybrid process calculus parser and translator to Modelica / Master thesis. – Univ. of Twente. – 2007. – 63 p.

Submitted for publication 2007 10 16

T. Krilavičius. Simulation of Mechatronic Systems using Behavioural Hybrid Process Calculus // Electronics and Electrical Engineering. – Kaunas: Technologija, 2008. – No. 1(81). – P. 45–48.

The growing interest in hybrid systems both in computer science and control theory has generated an interest in formalisms that can be used to specify and analyse such systems, systems that combine continuous-time and discrete behaviours. Simulation is one of the tools to obtain insight in dynamical systems behaviour. Its results provide information on performance of system, are helpful in detecting potential errors, and are handy in choosing adequate control strategies and parameters. We report a work in progress, a technique and a prototype for simulation of Behavioural Hybrid Process Calculus, an extension of process algebra suitable for the modelling and analysis of hybrid systems. Ill.1, bibl. 20. (in English; summaries in English, Russian and Lithuanian).

Т. Крилавичюс. Имитация мехатронных систем используя процесс алгебры для моделирования и анализа гибридных систем // Электроника и электротехника. – Каунас: Технология, 2008. – № 1(81). – С. 45–48.

Растущий интерес к гибридным системам в информатике и автоматике вызвал интерес к новым формализмам предназначенным для моделирования и имитации систем, описывающих аналоговое и дискретное поведение. Имитационное моделирование позволяет больше узнать о динамических системах. Результаты имитационного моделирования показывают эффективность системы, позволяют находить ошибки и подобрать подходящий метод и параметры управления. Мы описываем технику и прототип для имитационного моделирования процесс алгебры для моделирования и анализа гибридных систем “Behavioural Hybrid Process Calculus”. Ил.1, библ. 20 (на английском языке; рефераты на английском, русском и литовском яз.)

T. Krilavičius. Mechatroninių sistemų imitavimas naudojant BHPC imitacinio modeliavimo metodiką // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2008. – Nr. 1(81). – P. 45–48.

Augantis susidomėjimas hibridinėmis sistemomis informatikoje ir automatikoje skatina domėtis naujais formalizmais, skirtais hibridinėms sistemoms jungiančioms tolydų ir diskretų elgesį, modeliuoti ir imituoti. Imitavimas yra įrankis, leidžiantis sužinoti daugiau apie dinaminių sistemų elgseną. Imitavimo rezultatai suteikia informaciją apie sistemos efektyvumą, leidžia aptikti klaidas ir parinkti tinkamą valdymo strategiją ir parametrus. Straipsnyje pateikiama šiuo metu atliekamo tyrimo rezultatai: „Behavioural Hybrid Process Calculus“ (BHPC) imitacinio modeliavimo metodika ir įrankio prototipas. BHPC yra klasikinių procesų algebrų išplėtimas, skirtas hibridinių sistemų modeliavimui ir analizei. Il.1, bibl. 20 (anglų kalba; santraukos anglų, rusų ir lietuvių k.).

DOI: 10.5755/j02.eie.11032