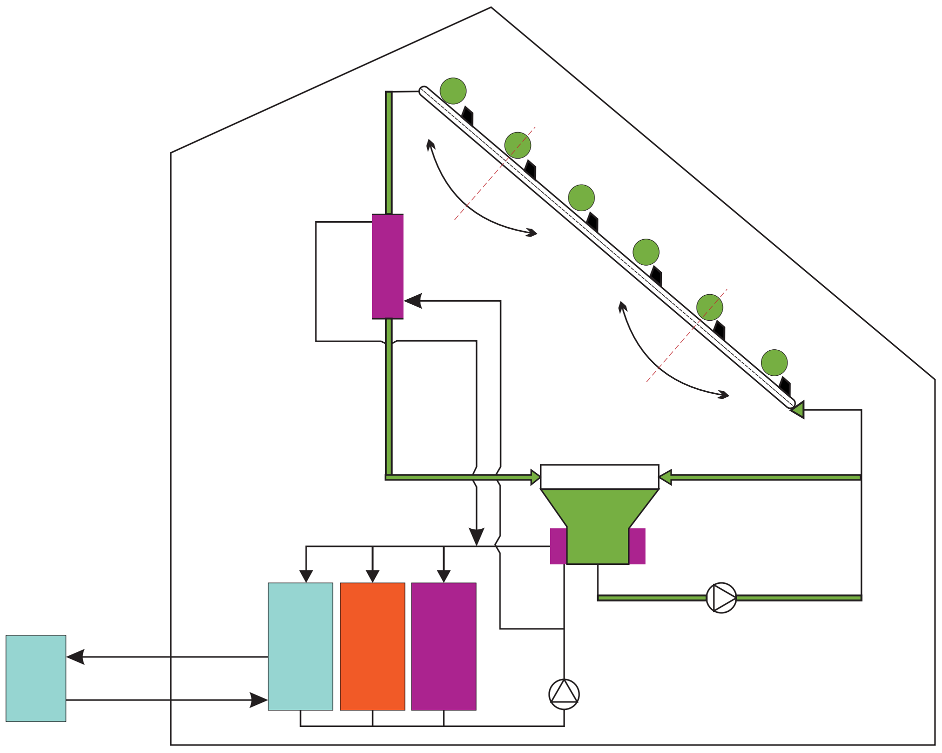
SNE SIMULATION NEWS EUROPE



Volume 17 Number 3/4

December 2007, ISSN 0929-2268

Journal on Developments and Trends in Modelling and Simulation

Membership Journal for Simulation Societies in EUROSIM

ARGESIM

Dear Readers,

With this SNE issue SNE 17/3-4 we have completed SNE's structural changes and changes in SNE's strategy. The new numbering with volumes is now fully introduced, and the series with SNE Special Issues is running now. These series continued in 2007 with the second special issue, emphasizing on 'Object-oriented and Structural-dynamic Modelling and Simulation' (SNE 17/2). This turned out to be of big interest, and we got more contributions than would fit into a special issue. So it was decided, to compile a second special issue on this subject in 2008 (SNE 18/2 'Object-oriented and Structural-dynamic Modelling and Simulation II'); the special issue on 'Validation and Verification' is postponed to 2009 (SNE 19/2).

The SNE Editorial Board is increasing, and it co-operates with IPCs from Simulation Conferences, to suggest conference papers for publication in SNE in extended and revised form. This issue publishes e.g. revised contributions from MATHMOD 2006 Conference (Vienna), and from Modelica 2006 Conference (Vienna). Furthermore, more space is available for Technical Notes (six in this issue), Short Notes (three in this issue) and Benchmark Notes (eleven in this issue). Technical Notes and Short Notes show the broad variety of modelling and simulation, from methods (object-oriented modelling with Siconos and with Multi-bond Graph Library, molecular dynamics, and dynamic programming) to applications (algal population modelling, boiler control simulation, transistor simulation, network performance modelling, and Yo-Yo – simulation). The eleven benchmark solutions show interesting classic and non-classic approaches and comparisons, e.g. solutions to ARGESIM Benchmark C18 'Neural Networks versus Transfer Functions - Identification of Nonlinear Systems' using MATLAB and MS Excel, four solutions to ARGESIM Benchmark C3'Analysis of Generalized Class-E Amplifier using object-oriented tools, and a VBA solution to the discrete ARGESIM Benchmark C8 'Canal-and-Lock System. The new layout and the double space for benchmark solutions have proven successful, increasing also the quality of solutions.

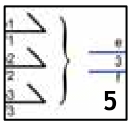
And last but not least, also the News Section shows a better information structure for EUROSIM Societies: general information tables and detailed recent information of societies). Because of delay of this issue and because of the unhappy fact, that some EUROSIM societies do not provide recent information, detailed reports are postponed to next issues.

We hope, readers enjoy all these innovations, and we thank all contributors, members of the editorial boards, and people of our ARGESIM staff for co-operation in producing this SNE issue.

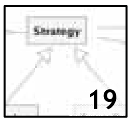
Felix Breitenecker, editor-in-chief, felix.breitenecker@tuwien.ac.at

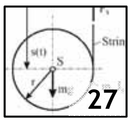
SNE 17/3-4 in Three Minutes

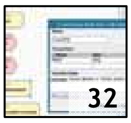
Impressum, Table of Contents: page 4

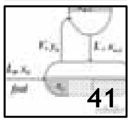

5 *The Modelica Multi-bond Graph Library* introduces bond graphs in Modelica as a powerful and intuitive way for modelling mechanical systems.

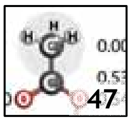
14 *Time Delay Model of Algal Population Growth in Tabular Photobioreactor* extends the Volterra-Lotka model to predict algal population under various physical conditions

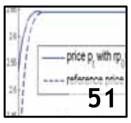

19 *Siconos: A Software Platform for Modeling, Simulation, Analysis and Control of NSDS* provides strategies for simulating Non-smooth Dynamical systems.

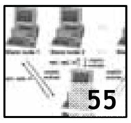

27 *Yo-yo Simulation on the Base of Analytical Treatment* derives an analytical solution for a yo-yo driven by a Dirac impulse and compares it to simulation results.

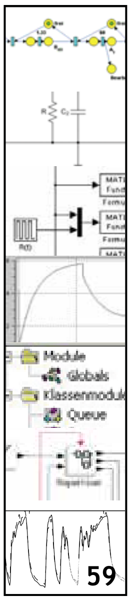

32 *Hybrid Performance Modeling and Prediction of Parallel and Distributed Computing System* introduces 'PerformanceProphet', a tool for prediction of computational costs.


41 *Optimal Multivariable Control Design Using Genetic Algorithms* presents a semi-batch rectification process and optimizes the quality of the product.


47 *Particle Dynamics Simulation at Atomic Scale: Molecular Dynamics* illustrates how to simulate thermodynamic properties of liquids through Molecular Dynamics (MD)


51 *Modelling Reference Price Effects – A discrete dynamic programming approach* provides a decision-support system for retail pricing and promotion planning.


55 *Parallel Transistor Level Simulation based on Parallel Equation Formulation implemented on a Beowulf Cluster* reports on a successful parallel simulation


59 ARGESIM Benchmarks section:

C2 'Flexible Assembly System'
- using MATLAB PetriSim toolbox 59
- using TaylorED 61

C3 'Analysis of a Class E Amplifier'
- using AnyLogic 6 63
- using Dymola 65
- using MATLAB/SimuLink 67
- using MOSILAB/Modelica 69

C5 'Two State Model'
- using MOSILAB 71

C8 'Canal-and-Lock System'
- using VBA 73

C13 'Crane Crab with Embedded Control'
- using Scicos 75

C18 'Identification of Nonlinear Systems'
- using MATLAB/NN toolbox 77
- using MS Excel/NeuralTools 79

81 *Book reviews, Journal reviews section* provides four Book reviews

7 pp. *EUROSIM short info* gives a short overview over EUROSIM and its national societies *Lukas Schuler* is introduced in the simulationists section

Table of Contents

The Modelica Multi-bond Graph Library
D. Zimmer, F. E. Cellier 5

Time Delay Model of Algal Population Growth in Tabular Photobioreactor
J. Fišer, J. Červený, P. Zítek 14

Siconos: A Software Platform for Modeling, Simulation, Analysis and Control of Nonsmooth Dynamical Systems. *V. Acary, F. Perignon* 19

Yo-yo Simulation on the Base of Analytical Treatment
R. Hohmann, L. Sperling 27

Hybrid Performance Modeling and Prediction of Parallel and Distributed Computing System
S. Pllana, S. Benkner, F. Breitenecker 32

Optimal Multivariable Control Design Using Genetic Algorithms
M. Atanasijević-Kunc, A. Belič, R. Karba 41

Particle Dynamics Simulation at Atomic Scale: Molecular Dynamics *L. D. Schuler* 47

Modelling Reference Price Effects – A discrete dynamic programming approach
L. Gimpl-Heersink, J. Wöckl, A. Taudes 51

Parallel Transistor Level Simulation based on Parallel Equation Formulation implemented on a Beowulf Cluster *B. Anđelković, V. Litovski* 55

A Petri Net approach to C2 'Flexible Assembly System' using the MATLAB PetriSimM toolbox
T. Löscher, Ch. Habersohn, G. Hinker 59

An Object-Oriented Solution to C2 'Flexible Assembly System' using TaylorED
T. Löscher, Yilin H., V. Legourski, O. Cevan 61

A System Dynamics Modelling Approach to C3 'Class E Amplifier' using AnyLogic *Š. Emrich, J. Asamer* 63

Comparison of Circuit Diagram Modelling and ODE Modelling for C3 'Class-E Amplifier' using Dymola
G. Zauner, P. Jahn, T. Polzer, A. Schuster 65

Simulation and Analysis of C3 'Class-E Amplifier' based on a Simulink Model with MATLAB functions
J. Asamer, G. Zauner 67

Comparison of Classical IF-clause Modelling and Statechart Modelling for C3 'Class-E amplifier' using MOSILAB/Modelica *G. Zauner, G. F. Kaunang* 69

Comparison of Classical IF-Clause Modelling and State Chart Modelling for C5 'Two State Model' in MOSILAB/Modelica *G. Zauner, G. F. Kaunang,* 71

A Directly Programmed Solution to C8 'Canal-and-Lock System' using Visual Basic for Applications
T. Löscher, J. Mosser 73

Modeling C13 'Crane and Embedded Control' with Hybrid System Approach using Scicos
M. Najafi, R. Nikoukhah 75

A Classic MATLAB-based Solution of C18 'Identification of Non-linear Dynamic Relations'
A. Belič 77

Solution to C18 'Identification of Nonlinear Dynamical Relations' using Excel with a Neural Networks Add-In *S. Blažič* 79

Book reviews 81

EUROSIM short info, industry information .... 7 pages

SNE Editorial Boards

Simulation News Europe (SNE) is advised by two Editorial Boards. The *SNE Editorial Board*, a scientific editorial board, is taking care on peer reviewing and handling of Technical Notes, Short Notes, Software Notes, Book and Journal Reviews, and of Comparison and Benchmark Notes (Notes Section). The *SNE News Editorial Board* (see News Section) is responsible for reports from EUROSIM, EUROSIM societies, International Societies, and Industry News.

SNE Editorial board

SNE Contact

SNE - Editors /ARGESIM
c/o Inst. f. Analysis and Scientific Computation
Vienna University of Technology
Wiedner Hauptstrasse 8-10, 1040 Vienna, AUSTRIA
Tel + 43 - 1- 58801-10115 or 11455, Fax – 42098
sne@argesim.org; www.argesim.org

Editorial Info – Impressum

SNE Simulation News Europe ISSN 1015-8685 (0929-2268).

Scope: Technical Notes and Short Notes on developments in modelling and simulation in various areas /application and theory) and on bechmarks for modelling and simulation, membership information for EUROSIM and Simulation Societies.

Editor-in-Chief: Felix Breitenecker, Inst. f. Analysis and Scientific Computing, Vienna University of Technology, Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria; Felix.Breitenecker@tuwien.ac.at

Layout: Markus Wallerberger, ARGESIM TU Vienna; markus.wallerberger@gmx.at

Printed by: Grafisches Zentrum, TU Vienna, Wiedner Hauptstrasse 8-10, 1040, Vienna, Austria

Publisher: ARGESIM/ASIM; ARGESIM, c/o Inst. for Scientific Computation, TU Vienna, Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria, and ASIM (German Simulation Society), c/o Wohlfartstr. 21b, 80939 Munchen

© ARGESIM/ASIM 2007

The Modelica Multi-bond Graph Library

Dirk Zimmer, François E. Cellier, ETH Zürich, Switzerland
{dzimmer, fcellier}@inf.ethz.ch

Bond graphs have established themselves as a reliable tool for modeling physical systems. Multi-bonds are a bond graphic extension that provides a general approach to modeling all kinds of multi-dimensional processes in continuous physical systems. This paper presents a Modelica library for modeling multi-bond graphs and their application to three-dimensional mechanical systems. A set of bond graphic models for ideal mechanical components is provided that enables a fully object-oriented modeling of mechanical systems. The wrapping of the bond graphic models and their representation by meaningful icons gives the mechanical models an intuitive appeal and makes them easy to use. The resulting mechanical systems can be efficiently simulated. Additionally, the continuous mechanical models were extended to hybrid models that allow discrete changes to be modeled that occur in mechanical system as a result of hard impacts.

1 Introduction

1.1 Introduction to Bond Graphs

If a physical system is subdivided into small components, we observe that these components all exhibit specific behavior with respect to power and energy: Certain components *store* energy like a thermal capacitance; other elements *dissipate* energy like a mechanical damper. An electric battery can be considered a *source* of energy. The power that is flowing between components is distributed along different types of junctions. This perspective offers a general modeling approach for physical systems: bond graphs: [3, 8].

Bond graphs are a domain-neutral modeling tool for continuous system modeling in the field of physics. The actual graph represents the power flows between the elements of a physical system. The edges of the graph are the bonds themselves. A bond is represented by a “harpoon” and carries two variables: the flow, *f*, written on the plain side of the bond, and the effort, *e*, denoted on the other side of the bond [3].

The product of effort and flow is defined to be power. Hence a bond is denoting a power flow from one vertex element to another.

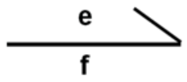


Figure 1: Representation of a bond

Domain	Effort	Flow
electrical	voltage (u)	current (i)
translational mechanics	force (f)	velocity (v)
rotational mechanics	torque (t)	angular velocity (ω)
acoustics / hydraulics	pressure (p)	volumetric flow (Φ)
thermodynamics	temperature (T)	entropy flow ($\acute{S}$)
chemical	chemical potential (μ)	molar flow (ν)

Table 1: Domain-specific effort/flow pairs

The assignment of effort and flow to a pair of physical variables determines the modeling domain. Table 1 below lists the effort/flow pairs for the most important physical domains.

The vertex elements are denoted by a mnemonic code corresponding to their behavior with respect to energy

Name	Code	*Equation*
resistance	**R**	$e = R \cdot f$
source of effort	**Se**	$e = e0$
source of flow	**Sf**	$f = f0$
capacitance	**C**	$f = C \cdot \mathrm{der}(e)$
inductance	**I**	$e = I \cdot \mathrm{der}(f)$
0-junction	**0**	all efforts equal sum $(f) = 0$
1-junction	**1**	all flows equal sum $(e) = 0$

Table 2: Mnemonic code of bondgraphic elements

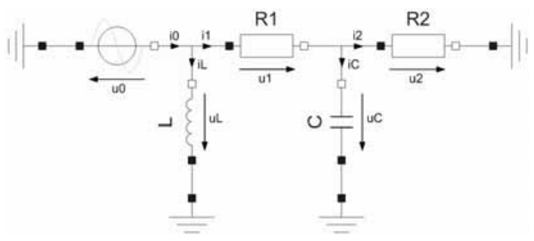


Figure 2: Schematic of electric circuit

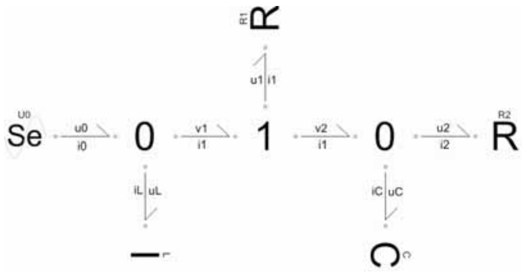


Figure 3: Bond graph of electric circuit

and power. Table 2 lists the most important bond graphic elements. The mnemonic code is borrowed from the electrical domain.

Using the bond graphic methodology, one can e.g. model the electric circuit of Figure 2 by the corresponding bond graph of Figure 3.

1.2 The Modelica BondLib

To conveniently model with bond graphs using Dymola, a Modelica library called BondLib [4] has been developed by F.E. Cellier and his students. Using this library, bond graphs can be created in an object-oriented fashion. The library provides a complete set of basic bondgraphic elements as atomic (equation) models. These can be composed graphically to form more complex composite models.

The basic bondgraphic elements and the bonds themselves can be placed on the screen by drag and drop. They can then easily be connected with each other. Further specifications can be added by means of parameter menus.

Although BondLib is a general Modelica library, its usage is strongly linked to the graphical modeling environment of Dymola [5]. Since bond graphs are a graphical modeling tool, it may be much less desirable to use this library in a purely alphanumerical modeling environment.

1.3 Introduction to Multi-bond Graphs

Multi-bond graphs (sometimes also called vector-bond graphs) are a vectorial extension of the regular

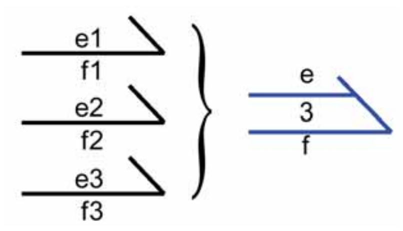


Figure 4: Composition of a multi-bond

bond graphs [2]. They are especially well suited for modeling multi-dimensional processes.

A multi-bond is composed of a certain number of bonds of either the same domain or at least closely related domains. It is represented by a (blue) double half-arrow as shown in Figure 4. The number shown at the center of the multi-bond denotes its cardinality, i.e., the number of individual bonds included.

This paper presents multi-bond graphs of mechanical systems, which is their primary application. However, multi-bond graphs can be used to model all kinds of multi-dimensional processes. Diffusion processes like heat distribution in a planar electric circuit may be another meaningful application. Yet another interesting field of application could be the modeling of chemical reaction dynamics [2]. Multi-bonds may also be applicable in the field of general relativity [7].

1.4 Advantages of Bondgraphic Modeling

Concerning the modeling of complex physical systems, bond graphs offer a suitable balance between specificity and generality. The interdisciplinary concept of energy and power flows creates a semantic level for bond graphs that is independent of the modeling domain. Thus, basic concepts of physics such as the first rule of thermodynamics can always be verified in a bond graph, independent of its application. This is particularly helpful for intra-domain models that operate in multiple energy domains. In addition, the semantic level helps the modeler avoid many types of modeling errors and find an adequate solution for his or her task. Modeling by equations is far more flexible and therefore leaves more room for mistakes.

Another advantage of bond graphs is their graphical approach to modeling. Relations can be expressed more naturally by two-dimensional drawings than in a one-dimensional equation code. Also the limitations of the screen (or drawing area) force the modeler to split his model into simple, easily understandable elements.

To us, bond graphs offer also a perfect approach to gaining a profound understanding of the basic principles covering all of physics and to organizing the knowledge concerning specific models. This makes bond graphs extremely valuable and useful for teaching purposes. However, bond graphs are a modeling tool like any other. Everything that can be modeled by bond graphs can also be modeled by other modeling paradigms. Some researchers will find bond graphs a convenient means to organizing their knowledge, whereas other researchers won't.

2 The MultiBondLib

The original BondLib only contains models for regular bond graphs. An additional library is needed to conveniently create models of multi-bond graphs. This paper presents a Modelica library for multi-bondgraphic modeling. It is called MultiBondLib and was designed to bear a strong resemblance to the existing BondLib in structure and composition. All users already familiar with BondLib should therefore be able to quickly acquaint themselves with the new MultiBondLib.

The MultiBondLib features also domain-specific sub-libraries for mechanical systems. These are introduced further on in Section 4. Although mechanics are the major field of application, the basic multi-bondgraphic elements provide a general solution to modeling all kinds of physical processes. These basic elements shall be briefly discussed in the following paragraphs.

Each vertex element of a regular bond graph has its multi-bondgraphic counterpart. The elementary equations of the basic bondgraphic elements remain the same. A transformation to the multi-bondgraphic terminology simply extends the scalar equations to vectorial form. The MultiBondLib provides these multi-bondgraphic counterparts and much more.

Just like the regular BondLib, the MultiBondLib features also causal bonds that are helpful for determining the computational causality of a model and for analyzing computational problems. A stroke denotes the side of the bond at which the flow vector is being computed. However, there is no need to ever use causal bonds in Modelica since the computational causality is determined automatically otherwise. Causal multi-bonds are also less convenient than their single-bondgraphic counterparts because the two causal multi-bonds do not cover all possible cases: mixed causality is possible as well, but a standard notation for mixed causality is missing and would probably be more confusing than helpful.

Figure 5: The two causal multi-bonds

In addition, MultiBondLib provides certain elements that are specific to multi-bond graphs. These elements handle the composition, decomposition and permutation of multi-bonds. Also special converter elements have been designed to allow a combination with the classic BondLib.

The cardinality of a multi-bond and its connected elements can be defined for each individual element separately. Consequently, it is possible to create models containing multi-bonds of different cardinality without any additional effort. Yet in many cases, the same cardinality is being used for the entire model. To afford a good usability in such cases, a default model has been developed. The default model is an outer model for all multi-bondgraphic components on the same level of the modeling hierarchy or below that defines the default cardinality of all underlying bondgraphic elements.

3 Mechanical MultiBond Graphs

In the mechanical domain, the bondgraphic effort is identified with the force, f, and the bondgraphic flow is identified with the velocity, v. The corresponding effort/flow pair of the rotational domain is torque, t, and angular velocity, ω. These assignments define the semantic meaning of the bondgraphic elements.

The inductance implements the fundamental equation $f = m \cdot dv / dt$ and represents storage of kinetic energy. The capacitance represents the storage of potential energy. It can be used to model a spring. A linear bondgraphic resistor models an ideal damper. All of these elements can be rigidly connected using a 1-junction, whereas a force interaction between two neighboring elements is modeled by a 0-junction.

Based upon this standard, various tools for the multi-bondgraphic modeling of mechanical systems have been developed since their invention by Breedveld in 1984 [2]. Unfortunately, most of these tools are incomplete, e. g. only suitable for the description of linear systems; and many are outdated.

To understand the specific difficulties that arise when using a bondgraphic approach to modeling a mechanical system, it is important to note that such systems often exhibit holonomic constraints, i.e., constraints resulting from the topology of the system. No two mechanical bodies can occupy the same space at the same time, and frequently, different mechanical bodies are related to each other by a distance constraint, e. g. they may be connected to each other by a mass-less bar. Holonomic constraints are constraints based on location. However, bond graphs operate on velocities and forces only. Hence mechanical systems cannot be modeled by bond graphs alone. There is a need for additional graphical modeling tools for expressing holonomic constraints between bodies.

The MultiBondLib offers such a link to other modeling tools through sensor elements. Sensors elements "measure" certain bondgraphic variables like effort, flow, momentum (integrated effort) or position (integrated flow). In the MultiBondLib, the signal emitted by the sensor elements is a-causal and simply represents an equality equation (not an assignment). Hence sensor elements are not limited to a single usage. They may serve a number of different purposes:

- to measure bondgraphic variables,
- to convert bondgraphic variables to non-bondgraphic signals, and
- to establish algebraic relationships between bondgraphic elements.

It is this latter form of usage that enables us to state holonomic constraint equations: The positional state is derived through sensor elements and influences the dynamical behavior through modulations. It is important to note that sensor elements as well as modulations are neutral with respect to power and energy. Hence the entire energy flow is described by the bond graph alone.

The following paragraphs present two examples that include the applications of these elements in the field of planar mechanical systems. In such a system, the world is restricted to two dimensions. Models for planar mechanic systems are therefore a lot simpler than three-dimensional models. They may serve as a good introduction to multi-dimensional mechanics since such models are much more easily understandable. One major simplification in planar mechanics is the fact that the rotational inertance, *J*, is a constant scalar for all possible orientations, because the axis of rotation is fixed. Thus everything can be comfortably

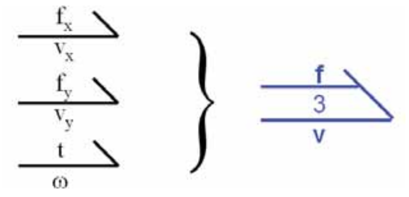


Figure 6: Composition of a planar multi-bond

computed using the coordinate vectors of the inertial system. There is no need for transformations between different coordinate systems as is the case in 3D-mechanics. All effort and flow variables of a planar mechanical bond graph can be conveniently resolved in the inertial system.

Hence planar mechanical systems can be described by multi-bond of cardinality three. Figure 6 depicts the planar mechanical multi-bond. The first two bonds belong to the translational domain, whereas the third bond belongs to the rotational domain. The effort vector of a planar multi-bond then is $(f_x, f_y, t)^{\mathrm{T}}$, whereas the corresponding flow vector is $(v_x, v_y, \omega)^{\mathrm{T}}$. Figure 7 presents the bondgraphic model of a simple planar pendulum.

The translational position is fixed at the revolute joint by a source of zero flow, *Sf*, of cardinality two. The revolute joint is free to rotate, i.e., it does not experience any torque. Hence it is modeled by a source of zero effort, *Se*, of cardinality one. The two multi-bonds are amalgamated to form a general multi-bond of planar mechanical systems of cardinality three.

The mass itself is represented by the 1-junction shown at the bottom of Figure 7. In a 1-junction, the flow variables (velocities) are equal, whereas the effort variables (forces) add up to zero. Hence the 1-junction represents the d'Alembert principle applied to the mass. The forces acting on the mass are the

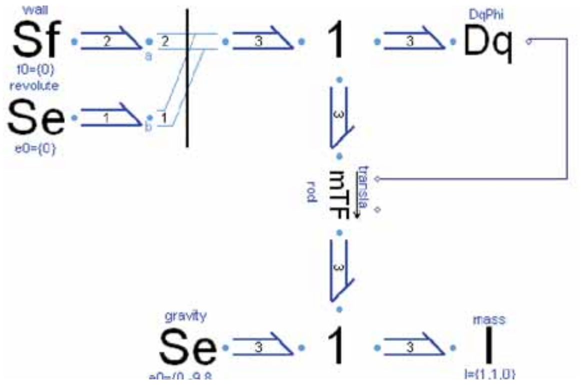


Figure 7: Bond graph of a planar pendulum

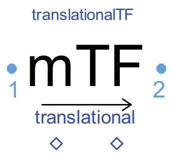


Figure 8: Icon of the modulated transformer. It provides two bondgraphic connectors (denoted by light blue circles) and two signal connectors (denoted by blue diamonds)

inertial force, *I*, and the gravitational force, which can be represented by another source of effort, *Se*, pulling in the negative *y*-direction.

The mass element is connected to the revolute joint by a mass-less rod describing a positional translation between the body element and the revolute joint. This rod is modeled by a modulated transformer element, *mTF*, that transforms the angular velocity into a translational velocity, and vice-versa.

This transformation is dependent on the current angle of the revolute joint. Therefore the transformer is modulated by the orientation angle of the revolute joint that is measured by the sensor element, *Dq*. The a-causal signal connecting the sensor element in combination with the transformer implements the holonomic constraint.

Let us take a closer look at the *mTF* element presented in Figure 8 to get a more profound understanding. The element inherits the bondgraphic effort and flow vectors (please remember the vector composition presented in Figure 6) $(\boldsymbol{e}_1, \boldsymbol{f}_1)$ and $(\boldsymbol{e}_2, \boldsymbol{f}_2)$, from its two bondgraphic connectors. These two connectors represent the hinges of a mass-less rod. Hence, the transformation between the variable pairs is specified by the parameter vector, ***d***, that represents the distance vector between the two hinges. A modulation of the transformation is determined by two signals: the current orientation φ and an optional elongation factor *ampl*. The actual model then defines an additional auxiliary variable, ***r***, and implements the balance equations of a lever:

$$\boldsymbol{r} = \begin{pmatrix} -\sin(\varphi) & \cos(\varphi) \\ -\cos(\varphi) & -\sin(\varphi) \end{pmatrix} \cdot \boldsymbol{d} \cdot ampl \qquad (1)$$

$$f_2[1;2] = f_1[1;2] + \boldsymbol{r}\, f_1[3]$$
$$f_2[3] = f_1[3]$$
$$e_1[1;2] = e_2[1;2]$$
$$e_2[3] = e_2[3] + \boldsymbol{r}^T e_2[1:2]$$

This modulated transformer, *mTF*, is a highly specialized element that hardly makes any sense outside the planar mechanical domain. Such specialized elements are thus provided in corresponding domain-specific sub-libraries, rather than listing them among the basic multi-bond graph elements of the MultiBondLib.

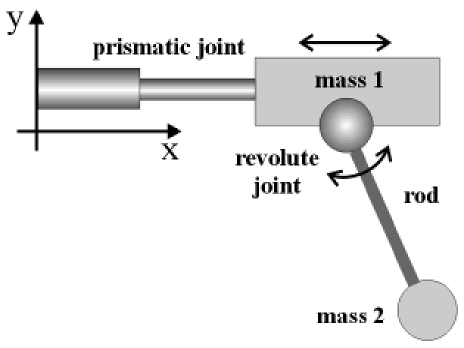


Figure 9: Schematic diagram of a crane crab

The large bond graph of Figure 10 represents the model of a second example: a simple model of a crane crab. The corresponding schematic diagram is presented in Figure 9.

Let us refrain from offering a detailed explanation of the model. Instead we shall take a look at the overall structure. The reader may observe the a-causal signals in Figure 10 that flow alongside the actual bond graph. These signals contain the variables for the current position and orientation. Whereas the multi-bond graph models the dynamics of the system, the signals handle the system's positional state.

Furthermore, multi-bond graphs of mechanical systems tend to become very large, since the sheer generality of the bondgraphic approach does not allow a more specific representation of larger mechanical elements. This makes the resulting bond graph hard to read and understand. It is therefore helpful to separate the bond graph into specific mechanically meaningful subparts as indicated by the gray rectangular frames of Figure 10. These subparts can then be represented by composite models that contain a wrapped version of the underlying subsystem multi-bond graph. The resulting model is presented in Figure 11 and is now easily understandable.

4 The Mechanical Sub-libraries

Two libraries for the modeling of mechanical systems have been developed, one for planar mechanics: "PlanarMechanics," and the other for 3D-Mechanics: "Mechanics3D". Both libraries are included as subpackages within the MultiBondLib. The following discussion will now focus on the components of the Mechanics3D library. However, most of the subse-

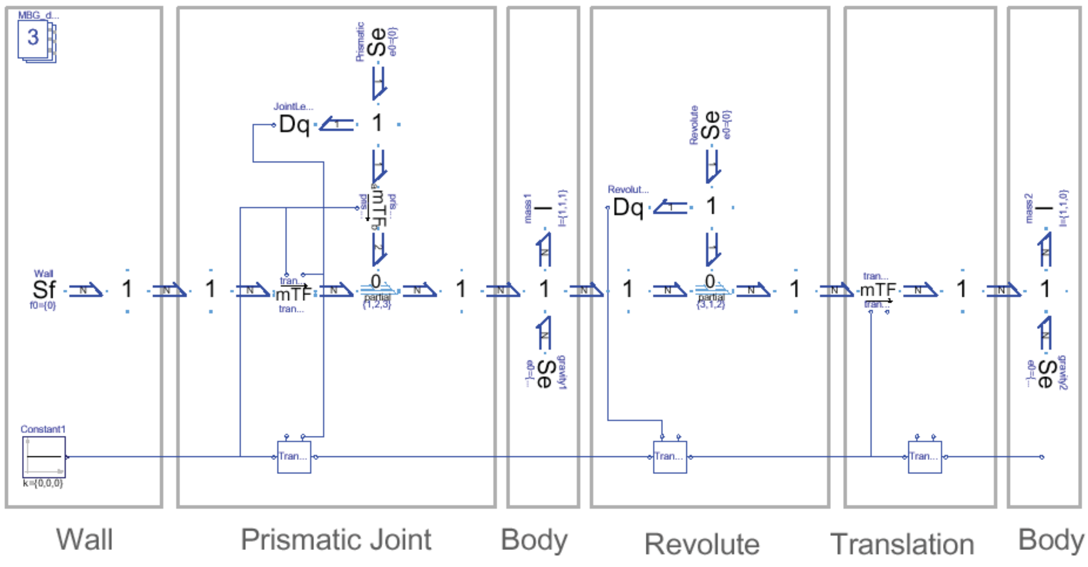


Figure 10: Multi-bond graph of a crane crab

quent descriptions hold for the planar mechanical library as well.

Similar to the Multi-body systems library of M. Otter [9], the Mechanics3D library offers models for an extensive set of mechanical components. It contains models for rigid parts such as bodies and rods, as well as models for different kinds of joints such as revolute joints or prismatic joints. A spring and a damper are typical examples of force elements for which models are offered in the library as well. In addition, models of ideal rolling objects such as wheels or marbles are provided. All of these elements can be further specified by parameter menus and feature a suitable animation.

Due to the similarity in structure and design, users of the standard Modelica MultiBody library will find the Mechanics3D library very easy to use. The Mechanics3D library represents both a subset and a superset of the MultiBody library. Yet in contrast to the standard MultiBody library, all mechanical models of the new Mechanics3D library are based upon wrapped bondgraphic models. A look inside the models reveals a bondgraphic explanation.

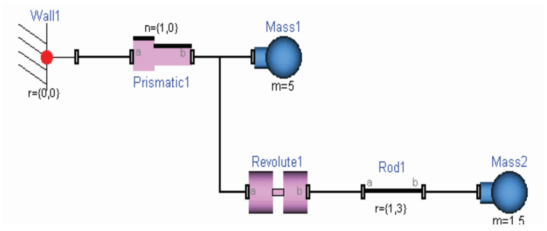


Figure 11: Multi-body diagram of a crane crab

4.1 Connector Types

In this section, we discuss the selection of connector variables for the mechanical components in Mechanics3D. This selection is defined by the process of wrapping. Therefore all bondgraphic variables have to be part of the connector. These are:

- the force vector, $\boldsymbol{f}$, (flow variables),
- the torque vector, $\boldsymbol{t}$, (flow variables),
- the velocity vector, $\boldsymbol{v}$, (potential variables),
- the angular velocity vector, $\boldsymbol{\omega}$, (potential variables);

where the variables of the rotational domain are resolved with respect to their corresponding body system. All other variables are resolved in the inertial system. Also the variables of the positional signals are part of the connector:

- the position vector, $\boldsymbol{x}$, (potential variables),
- the orientation matrix, $\boldsymbol{R}$, (potential variables).

Please note that the connector contains redundant information. The variables $\boldsymbol{v}$ and $\boldsymbol{\omega}$ are derivatives of $\boldsymbol{x}$ and $\boldsymbol{R}$. Also $\boldsymbol{R}$ itself is 3×3 orientation matrix and thus a redundant way of expressing the current orientation.

Although the Mechanics3D library and the MultiBody library are very similar, the Mechanics3D library defines its own slightly different connectors. Hence it is not possible to combine the components of these two libraries without additional tools.

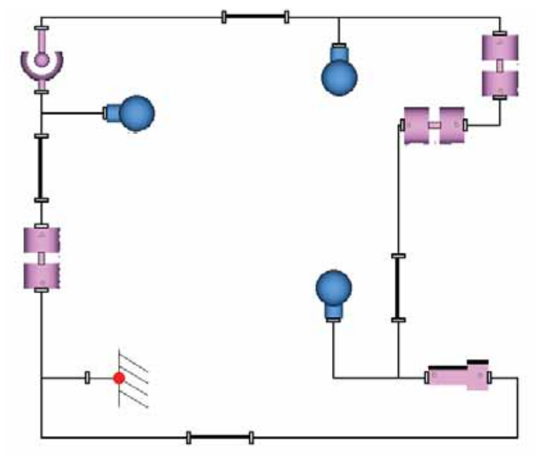


Figure 12: Multi-body diagram of a 3D kinematic loop

4.2 Kinematic Loops

The redundancy of the connector causes problems when components are connected in a circular fashion, as this is often the case in kinematic loops. A standard connection statement leads then to a singularity in the model. To overcome these difficulties, a second alternative connection statement is needed that contains only non-redundant information. For each kinematic loop, one standard connection has then to be replaced by its non-redundant counterpart.

However, the user of the library does not need to worry about these details, because the replacement is achieved automatically. To implement the necessary preprocessing of the model, a set of special Modelica functions[1] has been used. These are the same methods that have also been applied in the case of the standard MultiBody library (see [9]).

Figure 12 shows the Mechanics3D model of a 3D kinematic loop. No special cut joints have been used, and the elimination of redundant equations and redundant states takes place automatically. Whereas the automatic elimination algorithm works in most cases, planar loops within a 3D environment still require special treatment. A special cut joint has been developed for this purpose.

4.3 An Example: The Bicycle

This example was created by components of Mechanics3D and presents the model of an uncontrolled ideal rolling bicycle. Such a bicycle has seven degrees of freedom on the level of position and three degrees of freedom on the level of velocity. It is a partially stable system for a specific range of driving velocities. A nice description of the dynamic behavior of a bicycle can be found in a paper by K. Åström et al. [1]. The relevant parameter values for mass and geometry of this specific model are taken out of a paper by Schwab et al. [11]. The described bicycle is self-stabilizing for driving velocities between 4.3 ms^{-1} and 6.1 ms^{-1}.

Figure 13 depicts the multi-body diagram of the bicycle model. Although no closed loop is visible in the multi-body diagram, the model contains a closed kinematic loop, since both wheels are connected to the road. The Mechanics3D library offers an outer world3D model that is used in exactly the same fashion as the corresponding model of the standard MultiBody library.

The selected state variables are the cardan angles of the rear wheel and their derivatives. These three cardan angles represent the orientation on the plane, the lean of the rear frame, and the roll angle of the rear wheel. Additional state variables are the position of the rear wheel on the plane, the angle of the steering joint, and the angle of the front revolute joint. Each of the two wheels defines one holonomic constraint equation that prevents the bicycle from sinking into the road.

Figures 14 and 15 show the results of the simulation. The lean angle can be examined in a plot window, and the bicycle is nicely animated within Dymola.

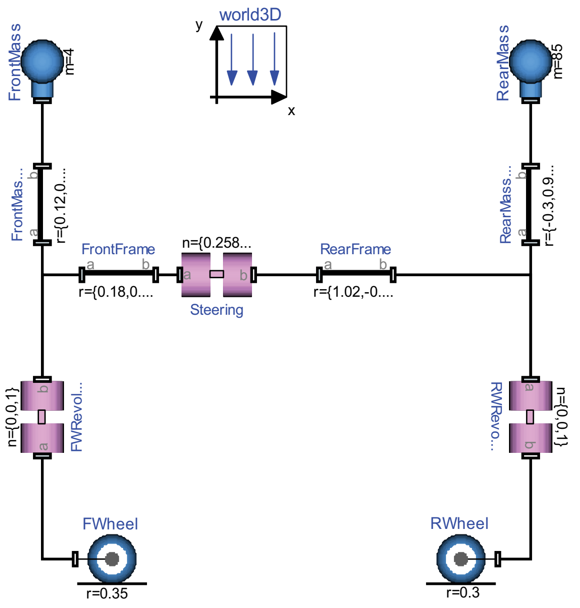


Figure 13: Multi-body diagram of a bicycle

[1] These are: defineRoot(), definePotentialRoot(), defineBranch(), isRoot(), and rooted().

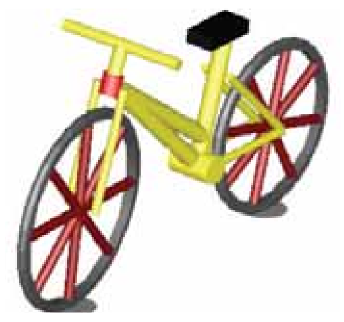


Figure 14: Animation model of the bicycle

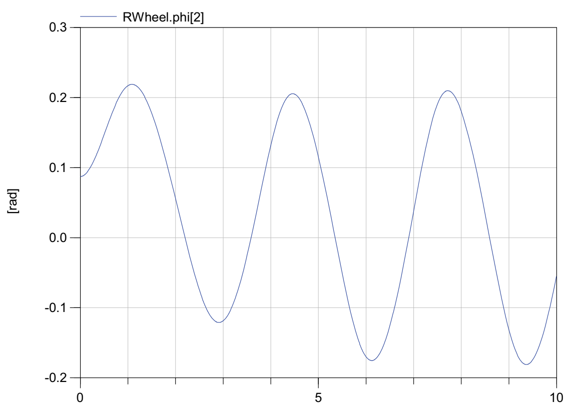


Figure 15: Plot of the lean angle

4.4 Run-time Efficiency

The selection of the state variables is of major importance for the efficiency of the resulting simulation. Usually this selection is automatically achieved by Dymola. Nevertheless variables can be suggested to the simulation program via the advanced Modelica attribute: “StateSelect”. These variables are the states of a joint’s relative position and velocity. Each joint is declaring state variables, unless there is a kinematic loop.

In the latter case, there are often many equivalent sets of state variables. Dymola uses the feature of dynamic state selection, where the state variables are chosen at run time. This offers the most flexible solution but leads to a slower simulation and to many additional equations that are often unnecessary. Hence it is strongly recommended to determine a static set of state variables manually in such a case whenever this is feasible.

All mechanical components that contain potential state variables have the option of a manual enforcement of their state variables via the parameter menu. To this end, the Boolean parameter “enforce-States” needs to be activated.

	MultiBody		**Mechanics3D**	
Experiment	Non-lin. eq.	Integr. steps	Non-lin. eq.	Integr. steps
Double Pendulum	0	549	0	549
Crane Crab	0	205	0	205
Gyroscopic Exp. with Quaternions	0	24438	0	25574
Planar Loop	2	372	2	372
Centrifugal	{2,2}	70	{2,2}	70
FourBar Loop*	5	446	5	625
Bicycle*	1	97	1	84

Table 3. Comparison of the two mechanical libraries

The Mechanics3D library and the standard MultiBody library contain elements for the same purpose that can be used in very similar ways. It is therefore easily possible to translate the model for a mechanical system from one library to the other. Both libraries are examined with respect to their run-time efficiency. The complexity of the resulting systems of equation and the computational effort of the simulation are compared for a given set of examples. The results of this examination are presented in Table 3. There is hardly any difference. In fact, the generated equations of both libraries are very similar. There is only one remarkable difference: In contrast to the Mechanics3D library, the translational velocity is not part of the connector variables of the standard MultiBody library. The elements are only connected by the translational position. The equations for the translational velocities are then derived (when necessary) by differentiation. In the bondgraphic models of the Mechanics3D library, these equations are explicitly stated.

Table 3 lists the sets of non-linear equations for a given set of experiments. The number of integration steps is counted for a simulation period of 10 seconds. The simulation method was Dassl with a tolerance of 10^{-4} A * indicates that the parameters of the experiment setup differ slightly between the two libraries.

5 Conclusions and Further Work

A Modelica library for convenient multi-bondgraphic modeling has been developed. It provides a general solution to modeling all kinds of multidimensional processes in continuous physical systems. Multi-bond graphs offer a good framework for

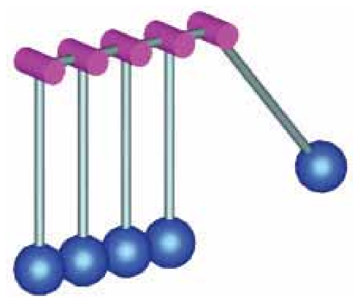


Figure 16: An implementation of Newton's cradle

modeling mechanical systems. The bondgraphic methodology proved to be powerful enough for modeling all important ideal subparts of mechanical systems accurately and efficiently.

The resulting libraries for mechanical systems PlanarMechanics and Mechanics3D provide an extensive set of component models. These domain-specific models have an intuitive appeal and are easy to use. They consist in wrapped bondgraphic models, and a closer look reveals a bondgraphic explanation of their behavior. Also quality and efficiency of the resulting solution are not impaired by the bondgraphic modeling technique. The selected state variables are chosen wisely, and the resulting systems of equations can be solved fast and accurately.

Furthermore, a third library for mechanics has been developed and included in the MultiBondLib. It is called "Mechanics3DwithImpulses" and represents an extension of the Mechanics3D library. The existing purely continuous mechanical models were extended by their corresponding impulse equations to hybrid models that contain an additional discrete event part.

This library was designed to model the behavior of mechanical systems in situations of hard ideal impacts. The type of mechanical system is thereby not limited at all. Impacts can be modeled between single objects, as well as between kinematic loops (e.g. a car suspension). The provided parameter set for the impact characteristics includes also specifications for elasticity and friction.

Sadly, the Modelica support for discrete event modeling [10] is not yet fully sufficient. Hence the resulting solution leads partially to models that are unnecessarily complicated in execution and design. Nevertheless the solution is fine and workable for small scale models. Figure 16 shows a simple example of a mechanical system modeled using the extended 3D library.

References

[1] Åström, K.J., R.E. Klein and A. Lennartsson, "Bicycle Dynamics and Control: Adapted Bicycles for Education and Research," *IEEE Control Systems Magazine,* **25**(4), pp.26-47, 2005.

[2] Breedveld, P.C., *Physical Systems Theory in Terms of Bond Graphs*, Ph.D. dissertation, University of Twente, The Netherlands, 1984.

[3] Cellier, F.E., *Continuous System Modeling*, Springer-Verlag, New York, 755p, 1991.

[4] Cellier, F.E. and A. Nebot, "The Modelica Bond Graph Library," *Proc. 4th International Modelica Conference*, Hamburg, Germany, Vol.1, pp.57-65, 2005.

[5] Dynasim AB, *Dymola Users' Manual*, Version 6.0, Lund, Sweden, 2006.

[6] Elmqvist, H., M. Otter and J. Díaz López, "Collision Handling for the Modelica MultiBody Library." *Proc. 4th International Modelica Conference,* Hamburg, Germany, Vol.1, pp.45-53, 2005.

[7] Gussn, N.M.N, *On the Bondgraphic Power Postulate and its Role in Interpreting the 1905 and 1915 Relativity Theories*, MS Thesis, Dept. of Electr. & Comp. Engr., University of Arizona, Tucson, AZ, 1994.

[8] Karnopp, D.C., D.L. Margolis and R.C. Rosenberg, *System Dynamics: Modeling and Simulation of Mechatronic Systems*, *4th edition*, John Wiley& Sons, New York, 576p, 2006.

[9] Otter, M., H. Elmqvist and S.E. Mattsson, "The New Modelica MultiBody Library," *Proc. 3rd International Modelica Conference*, Linköping, Sweden, pp.311-330, 2003.

[10] Otter, M., H. Elmqvist and S.E. Mattsson, "Hybrid Modeling in Modelica Based on the Synchronous Data Flow Principle," *Proc. IEEE International Symposium on Computer Aided Control System Design*, Hawaii, pp.151-157, 1999.

[11] Schwab, A.L., J.P. Meijaard and J.M. Papadopoulos, "Benchmark Results on the Linearized Equation of Motion of an Uncontrolled Bicycle" *KSME Journal of Mechanical Science and Technology*, **19**(1), pp.292-304, 2005.

[12] Zimmer, D., *A Modelica Library for MultiBond Graphs and its Application in 3D-Mechanics*. MS Thesis, ETH Zurich, Switzerland, 2006

Corresponding author: Dirk Zimmer
Institute of Computational Science
ETH Zurich,
CH-8092 Zurich, Switzerland
dzimmer@inf.ethz.ch

Accepted Modelica 2006 Conference, June 2006
Received: December 6, 2006
Accepted: January 20, 2007

Time Delay Model of Algal Population Growth in Tabular Photobioreactor

J. Fišer, J. Červený, P. Zítek, CTU Prague, Czech Republic

The paper deals with an approach to modelling the growth of an algal population cultivated in a bioreactor, applying a time delay system structure. With respect to the delayed causality between the light irradiance as well as supplying of nutrients on the one hand and the resulting growth on the other hand and also with regard to the life cycle of the cells, the model provides the possibility to predict the dry weight of algae as the output resulting from the influence of the physical conditions, particularly the incident light intensity and the water medium temperature as the inputs. Basically the model is designed as a functional extension of the Volterra-Lotka model of population growth and fits to the specific features of a pentroof tubular photobioreactor.

Introduction

There are two basic reasons why delays become a natural phenomenon in dynamical development and growth of biological systems [10]. First some delays appear between the nutrient supply and the resulting growth since the organisms store a part of nutrients for future utilization and also the utilized part of nutrients leads to a growth effect with a delayed causality. Secondly the life cycle of the cells, i.e. their ageing, endows the biological systems with a hereditary character. However, the time delays are usually introduced rather qualitatively in common population models and are not specifically tied to the biological parameters [5]. A tighter co-operation between the biologists and analytical researchers is therefore desirable. A widely applied population growth model has been developed on the basis of delayed Volterra-Lotka equations [5] while a more specific model respecting the age relationships can be found in [7]. A quite specific area of growth dynamics problems brings the issue of a tumour development, considered as an abnormal population growth [6]. Certain techniques in applying the population growth models to the issue of evolution of a cell population have been presented in [3]. These techniques can serve to find the best-fit parameters for a population model application by minimizing a cost function. A particular problem of this application to time delay models consists in difficulties with the derivatives of the cost function with respect to model parameters, including the time delays [2].

Three approaches to modelling of the cell population dynamics are observed in the system biology [1], [15]. Primarily the cell population growth is described by the division law of doubling the cells, usually referred to as exponential growth. The growth may also be regarded as a kind of sexually based reproduction within the population or as cloning the cells in the cultivated specimen of population [15]. Basically all these three approaches to the model design may result in an application of delayed Volterra-Lotka equations of population growth. Recent investigations of the ageing relations in a population have proved that the neutral delay differential equations (NDDE) are more fitting modelling tool than the retarded delay differential equations (RDDE). Hence the model structure has resulted in a neutral type of delayed Volterra-Lotka equation [9]. The virtue of the neutral form of the time delay model compared to the retarded one consists in the fact that the cell populations are not homogenous. The cells of various ages, maturity levels, activation status, duration of cell cycle, etc. are contained together in one population and the right-hand side derivative terms can properly express this reality. A consistent survey of modelling algae culture growth can be found in [9].

A specific issue in modelling the cell population growth consists in taking into account the influence of stressing factors on the population. The stress conditioned behaviour of population growth emerges regularly as a consequence of cell adaptation to variable environmental conditions, no matter whether these conditions suddenly change or persist far from a suitable state. The usual effect of this stress is a vanishing growth rate of population [11]. Particularly, a stress due to the high light irradiance, high or low temperature and nutrient stress are often observed. The cell adaptation to such unfavourable conditions is difficult to involve into the cell population models. Fortunately, most of the stress phenomena do not appear in properly controlled algae production in a photobioreactor.

1 Model Data Acquisition of Pentroof Tubular Photobioreactor

A basic objective of a continuously operating photobioreactor is to aim at a constant rate of population growth, also referred to as balanced growth rate. To reach the balanced growth rate the following prerequisites are to be provided: continuous food supply, sufficiently intensive and invariable lighting as well as environmental parameters in settled algae cultures. Under these conditions the average of the cell majority growth tends to be maintained constant. On the contrary in a batch operated photobioreactor the balanced growth is not attainable because of the cyclic character of the growth [4]. Data acquisition from the algae biomass production needed for the model formulation consists of the measurement of chlorophyll fluorescence emission, carbon dioxide concentration and off-line measurements of dry weight of algae or pigment concentration [15].

Algal culture cultivation is a complex biological process, able to produce a lot of bioactive compounds, which can be used primarily in pharmacy and cosmetics or as food additives. In the Czech Republic a novel pentroof tubular photobioreactor, designed for a continual algal biomass production, has been developed and built up at the Academy and University Centre in Nové Hrady with a financial support from EU, [13].

For the purposes of monitoring of algal population growth the chlorophyll fluorescence emission is acquired in an on-line regime while the dry weight of algae per volume unit is measured in discrete samples with a relatively long sampling period. Therefore the dry weight of algae per volume unit is not available for immediate controlling the algal cultivation. On the contrary the fluorescence emission measurement can provide the actual information about the influence of light irradiance on the algae population growth. In addition the fluorescence emission can also reveal possible stress behaviour in the population. If the stress behaviour has appeared it can be remedied by means of adequate shading the cultivation tubes and by a readjustment of focusing the Fresnel lenses (Fig. 1.).

The model variables are introduced as follows: the incident light intensity I on the external surface of the reactor tubes in [W· m^{-2}] and water medium temperature ϑ in [°C] are the inputs, and dry weight of algae per volume unit x in [kg·m^{-3}] is the output. The medium is a water solution of nourishing substances. Further variables as chlorophyll a, carbon dioxide and dissolved oxygen concentrations, or pigment concentrations turn out to be not indispensable for model formulation but their measurements are useful for the model validation. These variables need not be considered in the model structure if they are maintained in admissible limits during the experiments from which the measured data are acquired.

2 Time Delay Model of Algal Population

If the growth of the algal population is viewed as an asexual multiplying the model usually leads to a delayed Volterra-Lotka equation [9] and the model is then considered in the form

$$dx(t)/dt = r(x, I, \vartheta)\, x(t-d) \tag{1}$$

where $r(x, I, \vartheta)$ is an intrinsic overall rate of cell weight growth and d is a time delay between the occurring of incentives to the growth (food supply, lighting etc.) and the corresponding increase of algae amount. The intrinsic rate of growth results as a difference of two components

$$r(x, I, \vartheta) = \mu(x, I, \vartheta) - m(x, I, \vartheta) \tag{2}$$

where μ is the specific growth rate and m is the mortality rate of the algal population. Just the assessment of functions μ and m for measured I, ϑ and x is the key point to fitting the model (1). Mostly the temperature ϑ is considered constant and the functions μ and m (for healthy culture $\mu > m$) are usually given by the following formulae [8], [12], [14]

$$\mu(x, I) = \mu_{\max}\left(1 - e^{-\frac{I_a(x,I)}{I_m}}\right),\quad m(F_s) = m_{\max}\frac{K_s}{K_s + F_s} \tag{3}$$

where $Ia(x, I)$ is the average light intensity in the culture, Im is the light intensity level at which the cell growth is totally inhibited, Fs is the concentration of food supply and Ks is the half saturation constant for Fs. In addition $\mu_{\max}$ and $m_{\max}$ are the maximum specific growth rate and mortality rate, respectively. The average light intensity is usually considered as dependent on light intensity I and dry weight of algae x according to the following function

$$I_a(x, I) = \frac{I\,K_t}{C_t\,x}\left[\,1 - e^{-C_t x}\right] \tag{4}$$

where *Kt* is a specific parameter determining the irradiation effect in the photobioreactor tube and *Ct* is the biomass light absorption coefficient. The mentioned model does not possess the ability truly to express the existing influence of age distribution on population growth and therefore the following functional extension of model (1) is proposed

$$\frac{dx(t)}{dt} = \left[\mu(x,I,\vartheta) - \int_0^T dA(\tau)x(t-\tau) \right] x(t-d) + \left[\rho \int_0^T dB(\tau) \frac{dx(t-\tau)}{dt} - m(x,I,\vartheta) \right] x(t-d) \tag{5}$$

where $A(\tau)$, $B(\tau)$ are functions of delay distribution defined on the interval $\tau \in \langle 0, T \rangle$, T is the largest delay length, the meaning of μ and m remains unchanged, ρ is to distinguish the different levels of food consumption in a growing and mature populations. The distribution functions $A(\tau)$, $B(\tau)$ are positive over the entire interval of τ. While $A(\tau)$ applies to the changing space for the culture growth the distribution $B(\tau)$ expresses the influence of the ageing part of the population on the growth.

The distributed delays in (5) are considered in the general convolution form. The time delays in this equation are characterized not only by their distributed nature but also by the fact that equation (5) is of the so-called neutral type since the delays appear not only at *x* but also in the derivative term.

In available publications [5], [10] the specific rate of growth μ and mortality rate *m* are considered under only one single temperature. The temperature influence on the mortality rate is much weaker than that on the specific growth rate. Then, unlike the equation (2) the intrinsic rate of population growth for variable temperature can be expressed in the following form

$$r(x,I,\vartheta) = \mu_k(x,I)p(x,I,\vartheta) - m_k(x,I) \tag{6}$$

where *μk*, *mk* are the growth and mortality kinetics parameters for the reference temperature and *p* is an auxiliary function to express the changes of the growth rate with the temperature. This relationship has the following character: The lower is the biomass concentration *x* the lower light irradiance *I* can turn out to become inhibiting for the biomass growth. As regards the model (5) implementation nonzero initial conditions must be adjusted for purposes of modelling the algal population growth, corresponding to the initial biomass concentration of the algae culture.

Figure 1: Scheme of the pentroof tubular photo-bioreactor

3 Example of Modelling Pentroof Tubular Photobioreactor

The following example of photobioreactor brings a specific application of the above model (5) to a biotechnology implementation. The data used in this part originated from the measurements performed in the new solar pentroof photobioreactor in Nové Hrady, mentioned in the section 2. The scheme of the photobioreactor is in Figure 1.

The main components of the photobioreactor given in Fig. 1 are the following

1. linear Fresnel lenses
2. cultivation tubes
3. light heat absorbers
4. retention tank (degasser)
5. pump
6. water reservoir for heating and cooling
7. heat exchanger

The cultivated algae species in the pentroof tubular photobioreactor during the measurements was cyanobacteria *Spirulina platensis*. The parameters and delay distributions in (5) for this cultivation case have been assessed as follows

$$B(\tau) = \begin{cases} 0, & \tau \le \Phi_B \\ C_B \dfrac{\tau - \Phi_B}{T_B - \Phi_B}, & \tau \in (\Phi_B, T_B) \\ C_B, & \tau \ge T_B \end{cases} \tag{7}$$

where the parameters *ΦA*, *ΦB*, *TA*, *TB* are to be fitted for measured variables *I*, ϑ and *x*. On the basis of (3) the rates of the growth μ_k, and mortality m_k in relationship (6) result in the formulae

$$\begin{aligned}\mu_k &= \int_0^T dh_\mu(\tau)\,\mu_c(t) - e^{-\frac{I_a(x,I)}{I_m}} \\ m_k &= \int_0^T dh_m(\tau)\,x(t-\tau)\end{aligned} \tag{8}$$

where $I_a(x, I)$ is given by (4), μ_C is a constant parameter and $h_\mu(\tau)$, $h_m(\tau)$ are similar to (7) as follows

$$\begin{aligned} h_\mu(\tau) &= \begin{cases} 0, & \tau \le \Phi_\mu \\ C_\mu \dfrac{\tau - \Phi_\mu}{T_\mu - \Phi_\mu}, & \tau \in (\Phi_\mu, T_\mu) \\ C_\mu, & \tau \ge T_\mu \end{cases} \\ h_m(\tau) &= \begin{cases} 0, & \tau \le \Phi_m \\ C_m \dfrac{\tau - \Phi_m}{T_m - \Phi_m}, & \tau \in (\Phi_m, T_m) \\ C_m, & \tau \ge T_m \end{cases} \end{aligned} \tag{9}$$

Parameters Φ_μ, T_μ, C_μ, Φ_m, T_m, C_m were identified from the measurements of the variables I, ϑ and x, plotted in Fig. 2, 3. Later on, parameters K_t, C_t and I_m in (4) are given by the photobioreactor itself and the species of algae. The auxiliary function p is left to be determined. Its form has been obtained as follows

$$p = \begin{cases} 0.15, & 10000 \le \frac{I}{x} \text{ and } 20°C < \vartheta \le 36°C \\ 0.5, & 3000 \le \frac{I}{x} < 10000 \text{ and } 20°C < \vartheta \le 30°C \\ 0.75, & 3000 \le \frac{I}{x} < 10000 \text{ and } 30°C < \vartheta \le 36°C \\ 1, & \frac{I}{x} < 3000 \text{ and } 30°C < \vartheta \le 36°C \end{cases} \tag{10}$$

The water medium temperature was maintained within the interval $\vartheta \in \langle 20, 36 \rangle °C$. For a specimen of input variables I and ϑ in Figure 2 and the corresponding output biomass concentration x in Figure 3 the model (5) parameters have been assessed as listed under the Figure 3. Simultaneously the model output responses are compared with the available measurements.

The maximum specific growth rate µmax and mortality rate mmax in (3) resulted from the measurements in Nové Hrady and the distribution functions in (9) were fitted to their values. The pure time delay d in model (5) resulted in a small value and was therefore neglected.

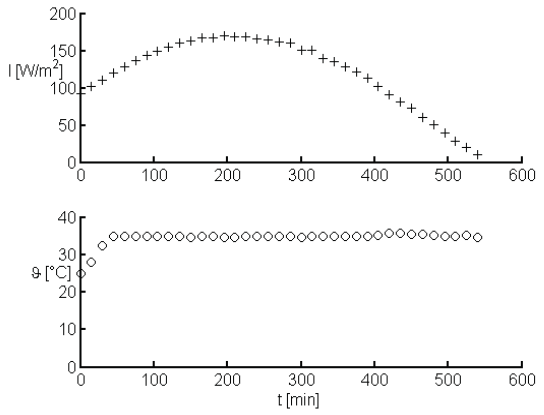


Figure 2: Inputs to model (5): incident light intensity and water medium temperature

4 Conclusions

A time delay nonlinear model of the algal population growth has been proposed, structured on the basis of delay differential equation of the neutral type. The distributed delay terms expressed as convolutions in the model allow to respect the impact of ageing of algae population on its growth. Specifically the model has been developed for an application to the tubular photobioreactor designed to operate with a balanced growth rate of algae population. To facilitate the identification of the model the delay distribution functions in the convolution terms are considered as simple ramp functions. The dependence of both intrinsic rates of growth and mortality on the light irradiation and food supply is expressed by already introduced nonlinear functions (μ and m) while the influence of temperature change is included by means of the auxiliary function p. The model identified on the basis of the measurements in Nové Hrady proved a good accordance of the model with the measurements on the tubular photobioreactor.

Acknowledgement

This work was supported by the Ministry of Education of the Czech Republic under **Project 1M0567** and **Project SM6007665808** and also by the Czech Science Foundation, **Project 522/05/P276**.

References

[1] Arino, O., Hbid, M.L., Ait Dads, E., *Delay Differential Equations and Applications*. Springer, Dodrecht, 2006

[2] Baker Ch. T. H., Bocharov G. A. and Rihan F. A., *A Report on the Use of Delay Differential Equations in Numerical Modelling in the Bioscience*. Manchester

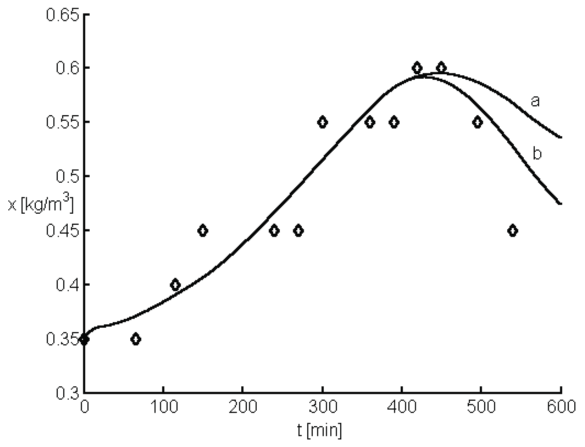


Figure 3: Measured biomass concentration compared with model (5) responses on the incident light intensity and the water medium temperature under model parameters $\rho = 0.5$, $d = 0$,

$\Phi A = 5$ min,	$TA = 15$ min,	$CA = 0.0075$,
$\Phi B = 5$ min,	$TB = 6$ min,	$CB = 0.0075$,
$\Phi\mu = 140$ min,	$T\mu = 360$ min,	$C\mu = 360$ min,
$Fm = 400$ min,	$Tm = 520$ min,	$Cm = 0.003$,
$Ct = 336$,	$Kt = 0.6674$,	$Im = 193$ W/m^2

and function $\mu c = 312.5\ (t)$.
a) $h\mu(\tau) = 0$, b) $h\mu(\tau) \neq 0$

Centre for Computational Mathematics, University of Manchester UMIST, Numerical Analysis Report No. 343, 1999, pp. 46

[3] Baker Ch. T. H., Bocharov G. A., Paul C. A. H. and Rihan F. A., *Modelling and Analysis of Time-lags in Some Basic Patterns of Cell Proliferation*. J. Math. Biol., 37 (1998), 341-371

[4] Cullen, J., *Factors Limiting Primary Productivity in the Sea. Environ*. Sci. Res., 43 (1992), 69-88.

[5] Cushing J. M., *An Introduction to Structured Population Dynamics*. Regional Conference Series in Applied Mathematics, Vol. 71, SIAM, Philadelphia 1993

[6] Davison E. J., *The modelling and behaviour of normal and abnormal cell growth*. Proc. 6th Triennial World Congress of IFAC, Boston, MA, 1975, 24-30

[7] Iannelli M., *Mathematical Theory of Age-structured Population Dynamics*. Applied Mathematics Monographs, Vol. 7, Giardini Editori e Stampatori, Pisa 1994

[8] Li, J., Xu N. S. and Su W. W., *Online Estimation of Stirred-tank Microalgal Photobioreactor Cultures Based on Dissolved Oxygen Measurement*. Biochemical Eng. J., 14 (2003), 51-65

[9] Kuang, Y., *Delay Differential Equations, with Applications in Population Dynamics*. In: Mathematics in Science and Eng., 191, (Ed.: Ames, W. F.) Academic Press, Inc., San Diego, 1993.

[10] MacDonald N., *Biological Delay System: Linear Stability Theory*. Cambridge University Press, Cambridge 1989

[11] MacIntyre H. L. and Cullen J. J., *Using Cultures to Investigate the Physiological Ecology of Microalgae*. In: Algal Culturing Techniques, (Ed.: Andersen, R. A.) Elsevier Academic Press, London, 2005, 287 - 326.

[12] Monod J., *The Growth of Bacterial Cultures*. Annual Rev. Microbiol., 3 (1949), 371-394

[13] PHARE, *Cross-border Co-operation Programme for the Czech Republic*, Project No. Cz 01.11.01.01, BIOTECHNOLOGY CENTER, Nové Hrady, Czech Republic.

[14] Rorrer G. L. and Mullikin R. K., *Modelling and simulation of a tubular recycle photobioreactor for macroalgal cell suspension cultures*. Chemical Engineering Science, 54 (1999), 3153-3162

[15] Wood, A. M., Everroad R. C. and Wingard L. M., *Measuring Growth Rates in Microalgal Cultures*. In: Algal Culturing Techniques, (Ed.: Andersen, R. A.) Elsevier Academic Press, London, 2005, 269 - 286.

Corresponding author: J. Fišer,
Centre for Applied Cybernetics
Czech Technical University in Prague
Technická 4, 166 27 Praha 6, Czech Republic
Jaromir.Fiser@fs.cvut.cz

Accepted MATHMOD 2006, January 2006
Received: June 6, 2006
Revised: February 10, 2007
Accepted: March 10, 2007

Siconos: A Software Platform for Modeling, Simulation, Analysis and Control of Nonsmooth Dynamical Systems

Vincent Acary, Franck Perignon, INRIA Rhône-Alpes, France
vincent.acary@inrialpes.fr, franck.perignon@inrialpes.fr

In this paper, a brief overview of the so-called nonsmooth dynamical systems and their links with hybrid systems is first proposed. In particular, Lagrangian mechanical systems with contact and friction or electrical systems with ideal components (diodes, MOS transistors ...) belong to this framework. Secondly, the Siconos software, dedicated to simulation of such systems, is presented, starting from its architecture and the way NSDS are modeled within the platform. To conclude, three representative samples are shown in order to illustrate the Siconos platform abilities.

1 A short introduction to Non-Smooth Dynamical Systems (NSDS)

The Non Smooth Dynamical Systems (NSDS) are a special kind of dynamical systems characterized by the non-smoothness of their time-evolution and of their constitutive equations. The instants of discontinuity of the state or its derivatives can be viewed as events for which the system changes in structure, in other terms as a transition. In this way, a NSDS combines features of continuous dynamical systems with the characteristics of finite automata. Thus, as a mixture of time-continuous dynamics and discrete systems, the NSDS can be viewed as a subclass of hybrid systems.

Like the term "hybrid", the “nonsmooth” one is not a precise definition, in the sense that it is not restrictive enough. A better way to define a coherent class of dynamical systems is to consider the mathematical nature of their solutions depending on the chosen formulation. This introduction aims at giving a flavor of mathematical properties and resulting numerical consequences, which are shared by NSDS. The term “non-smooth” is partly inherited from the extensive use of a well-recognized mathematical theory: the non-smooth analysis [2]. Since they are specialization of hybrid dynamical systems, new mathematical results for NSDS can be derived and new efficient simulation tools can be designed. The role of the Siconos platform is then to take advantage of these properties in order to provide general modeling and simulation tools for NSDS.

1.1 The example of constrained dynamics

A standard example of NSDS is a dynamical system of the form:

$$\dot{x} = f(x,t), \quad x \in \mathbb{R}^n,\ t \in [0,T] \qquad (1)$$

subjected to a set of constraints on its state (the inequality is to be understood component-wise):

$$y = h(x) = [h_\alpha(x), \alpha = 1...m]^T \geq 0 \qquad (2)$$

The constraints (2) are usually enforced by an external input, say a multiplier $\lambda \in \mathbb{R}^m$ through an input function g such that

$$\begin{cases} g:\ \lambda \in \mathbb{R}^m \mapsto g(\lambda) \in \mathbb{R}^n \\ \dot{x} = f(x,t) + g(\lambda) \end{cases} \qquad (3)$$

Finally, in order to complete the system, additional modeling information is needed. Particularly, two laws are of utmost importance:

a) A generalized equation [10] between the output w and the multiplier λ, denoted by the following inclusion:

$$0 \in F(y,\lambda) + Q(y,\lambda) \qquad (4)$$

where $F:\mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ is assumed to be continuously differentiable and $Q:\mathbb{R}^{m\times n} \rightsquigarrow \mathbb{R}^{m\times n}$ is a multi valued mapping with a closed graph. In simple cases, a complementarity condition is usually introduced for (4):

$$0 \leq y \perp \lambda \geq 0 \qquad (5)$$

b) A reinitialization mapping or an impact law defining the state of the system after a non smooth event:

$$x(t^+) = \mathcal{F}(x(t^-),t) \qquad (6)$$

A great number of dynamical systems with non-smooth evolution can be casted into the previous formulation. To cite a few of them, we can mention differential inclusions, projected dynamical systems, piecewise affine systems and linear complementarity systems. For a comprehensive review of NSDS, we refer to the following monographs, [1] and [12].

Lagrangian dynamical systems

To shed more light on what NSDS can be, we can consider for instance the well known case of the Lagrangian mechanical systems with unilateral constraints, which model the dynamics of finite dimensional mechanical systems with contact. Let us consider a p-dimensional Lagrangian system in a configuration manifold $\mathcal{M}$, parameterized by a set of p generalized coordinates denoted by $q \in \mathcal{M}$. The matrix $M(q)$ is the mass and $F(q,\dot{q},t)$ the set of forces acting upon the system. Usually, the unilateral constraints are written in the coordinates as:

$$h_\alpha(q(t)) \geq 0, \quad \alpha = 1\ldots\nu, \tag{7}$$

defining an admissible set for the system:

$$\Phi(t) = \{z(t) \in \mathcal{M}, \quad h_\alpha(z(t)) \geq 0, \alpha = 1\ldots\nu\}. \tag{8}$$

With sufficient regularity in time, the Lagrange equations are:

$$M(q)\ddot{q} + F(q,\dot{q},t) = \sum_{\alpha=1}^{\nu} \nabla h_\alpha(q)\lambda_\alpha \tag{9}$$

where λ_α is the set of the Lagrange multipliers associated with the constraints $h_\alpha(q(t))$ trough a complimentary condition:

$$0 \leq h_\alpha(q) \perp \lambda_\alpha \geq 0 \tag{10}$$

In this case, the output function h is defined in terms of the coordinates q and the output function g is related to this function by:

$$g(\lambda) = \begin{bmatrix} 0 \\ \sum_{\alpha=1}^{\nu} \nabla h_\alpha(q)\lambda_\alpha \end{bmatrix} \tag{11}$$

On one side the Lagrangian dynamical system can be understood as an hybrid system with 2^ν modes, a mode being defined by the fact that each constraint α is active or not. In each mode, the solution is assumed to be smooth with discontinuities arising at transitions between two modes. On the other side, it can be considered as a single NSDS. Then its solution is defined as a global one, possibly non smooth, containing the instants of discontinuity. Usually, for Lagrangian systems, the coordinates are considered as absolutely continuous functions of time, the velocities as functions of bounded variations and accelerations as measures.

Linear complementary systems

In [5], the authors study the so-called Linear Complementarity Systems (LCS) defined by:

$$\begin{cases} \dot{x} = Ax + B\lambda \\ y = Cx + D\lambda \\ 0 \leq w \perp \lambda \geq 0 \end{cases} \tag{12}$$

In this type of systems, a linear output y of a linear dynamical system is defined as the complementary variable of an input λ. The linear complementarity condition can model for instance the behavior of an ideal diode. If the output y represents the current through the diode, this condition states that this current must be positive. The multiplier λ is then the opposite of the voltage across the diode. The graph of this relation is illustrated on Figure 1.

1.2 Comparison between the hybrid and the non-smooth approaches of NSDS

From the point of view of discrete systems, hybrid systems can be modeled as infinite-state transition systems [6]. For simple dynamics (constant or linear) and small systems (up to 100 degrees of freedom), the hybrid approach allows us to exploits the widespread analysis techniques for finite-state systems, such as the verification techniques to check some fundamental properties. For larger systems with fully nonlinear dynamics, it seems that these discrete techniques no longer apply.

In the case of NSDS, the mathematical properties of the solution and the associated numerical techniques allow the simulation and the analysis of large systems to be performed. We claim that the continuous approach is more efficient from the mathematical and the numerical point of view, rather than an event-driven or a discrete approach.

On the mathematical point of view, a NSDS can be considered as a unique time-continuous dynamical

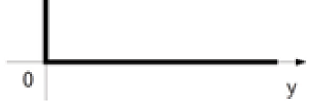


Figure 1. Complementarity condition as a constitutive law for an ideal diode

system which can encounter discontinuities and reinitializations. That leads to the definition of global solution of such systems in a class of appropriate functions. The case of the constrained Lagrangian dynamics leads for example to the formulation of the dynamics in terms of measure differential inclusions [11, 8, 7] valid on the continuous part of the evolution as well as at the events. Such types of solutions and formulations can comprised complex sequences of events, like concurrent events or accumulation (Zeno), together with mathematical results such as global existence and uniqueness.

On the numerical point of view, a non smooth approach of hybrid systems exploits the previous notion of solution defined everywhere in time. This fact leads to a design of powerful time–stepping schemes without explicit event handling procedure. For the case of the Lagrangian dynamics, the measure differential inclusion is evaluated on a fixed time interval and structural changes of the dynamics are taken into account in a weak sense. Contrary to event-driven schemes, such time-stepping are proved to be convergent even in presence of events accumulations.

Another advantage of the nonsmooth approach of NSDS is the algebraic formulation of certain class of state transitions at events. To shed more light on this aspect, the simple example of an ideal diode can be taken. This behavior can be modeled as a pure logical component thanks to an ‘if’ statement as in Modelica [4]. Indeed recognizing that the curve of the graph in Figure 1 can be parameterized by a parameter *s*, we can define the following Modelica script:

```
off = s < 0
  λ = if  off  then  -s  else 0
  y = if  off  then  0   else s
```

Same representations can be performed with ideal switches, piece-wise linear model of MOS transistors. The main difficulties to view systems with ideal components this way is that for each new boolean variable like off, two modes of the hybrid dynamical system are possible. If we introduce n-boolean variables, in the worst case, 2^n modes have to be checked. Therefore the problem complexity is exponential.

On the contrary, in the nonsmooth approach the discretized problem at each step can be reformulated as a Linear Complementarity Problem (LCP)[3] of the form:

$$\begin{cases} w = Mz + q \\ 0 \le w \perp z \ge 0 \end{cases} \tag{13}$$

Under some usual assumptions on the matrix M, (positiveness, n-step property), and on the vector q, numerical algorithms can be used with polynomial complexity, avoiding an exhaustive enumerative verification of each modes at exponential time [9].

To conclude, on the mathematical point of view, the non smooth framework yields precise definitions of solutions together with uniqueness and existence results under appropriate assumptions. On the numerical point of view, the use of specific algorithms (time–stepping schemes, LCP solvers with polynomial complexity) leads to an efficient simulation environment.

2 NSDS in Siconos: problem formulation and software architecture

2.1 Overview of Siconos software

Siconos software is mainly dedicated to modeling and simulation of NSDS. It aims at providing a general and common tool for non smooth problems present in various scientific fields: applied Mathematics, Mechanics, Robotics, Electrical networks and so on. However, the platform is not supposed to re-implement the existing dedicated tools already used for the modeling of specific systems, but to possibly integrate them. For instance, strong collaborations exists with HuMAns[1] (humanoid motion modeling and control) or LMGC90[2] (multi-body contact mechanics) softwares.

Siconos software development is part of a European project involving different research teams (More details on `siconos.inrialpes.fr`), but is mainly developed in Bipop team at INRIA Rhônes-Alpes in Grenoble.

Siconos is a free software, under GPL GNU license, available on the Gforge web pages of the project, where one can also find documentation, support and all that sort of utilities: `http://siconos.gforge.-inria.fr/`.

Siconos is composed in three main parts: Numerics, Kernel and Front-End, as represent on figure 2.a below.

[1] `http://bipop.inrialpes.fr/software/humans/`
[2] `http://www.lmgc.univ-montp2.fr/~dubois/LMGC90/`

The Front-End provides interfaces with some specific command-languages such as Python or Scilab, this to supply more pleasant and easy-access tools for users, during pre and post treatment.

The Numerics part holds all low-level lgorithms, to compute basic well-identified problems (ordinary differential equations, LCP, QP solvers, Blas-Lapack linear algebra routines ...).

Finally, the Kernel is the core of the software, providing high level description of the studied system and numerical solving strategies. It is fully written in C++, and mainly composed of two rather distinct parts, modeling and simulation, that will be described more in details below. The whole dependencies among Kernel parts are fully depicted on figure 2.b. The Utils module contains tools, mainly to handle classical objects such as matrices or vectors. The Input-output module concerns objects for data management in XML format, thanks to the libxml2 library. Finally, a plug-in system is also available, essentially to allow user to provide his own computation methods for some specific functions (vector field of a dynamical system, mass ...), this without having to re-compile the whole platform. Moreover, the platform is designed in a way that allows user to add dedicated modules through object registration and factories mechanisms.

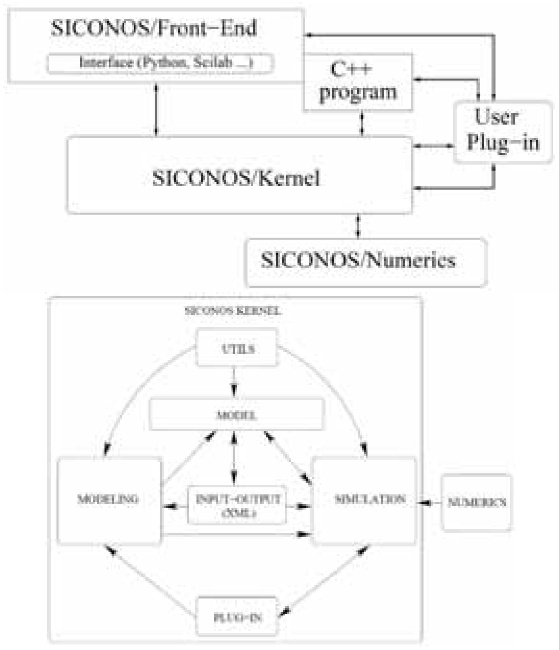


Figure 2: (a) General design of Siconos software, (b) Kernel component dependencies

2.2 NSDS modelling in Siconos software

As explained in Section 1, a non smooth dynamical system is a set of dynamical systems that interact altogether in a non smooth way. This is the role of Kernel modeling part to provide tools for these systems description.

Main objects are the pre-mentioned `DynamicalSystem`, `Relation` and `NonSmoothLaw`, both embedded in `Interaction`. To enlighten this point let us consider the example of a column of two spherical balls, in contact or not, falling down to the ground. Each ball constitutes a `DynamicalSystem`, including ordinary differential equations to described its dynamics, i.e. trajectories, initial conditions and other useful variables. Then, two interactions are possible: one between the balls and another between the ground and the lowest ball. A Siconos `Interaction` contains a list of dynamical systems (from one to any number, two in the present case), a relation that linked global and local variables and a nonsmooth law to describe the way systems interact. For the ball column case, the relation states that distance between two balls, and between lowest ball and ground, has to remain positive, and the law can be Newton impact one (see 2.2.c below for more details).

A reduced diagram of Siconos Kernel is presented on figure 3.a, which makes clearer various links between previous mentioned objects. In following paragraphs, various types of systems, relations and laws will be described more in details.

Dynamical systems

The most general case available in the platform is a first order system of the form:

$$\dot{x} = f(x,\dot{x},t) + T(x)u(x,\dot{x},t) + r \qquad (14)$$

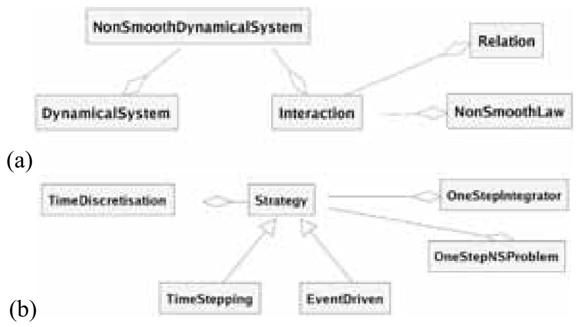


Figure 3: Simplified class diagram for kernel (a) modeling part, (b) simulation part.

where r is the non smooth part (typically contact forces for mechanical systems). (Note that to lighten equations, we will write x rather than $x(t)$). The terms Tu introduce the control variable into the system. All other dynamical systems in the software derived from the one above. They are:

- Linear Dynamical Systems:
$$\dot{x} = A(t)x + Tu(t) + b(t) + r \tag{15}$$
- Lagrangian (second order) systems, usual in mechanical problems:
$$M(q)\ddot{q} + NNL(q,\dot{q}) = F_{\text{int}}(\dot{q},q,t) + F_{\text{ext}}(t) + r \tag{16}$$
where q denotes generalized coordinates, M the mass matrix, NNL the nonlinear inertia operator, F_{int} the internal, nonlinear forces and F_{ext} the external forces, depending only on time.
- Lagrangian Linear Time Invariant systems:
$$M\ddot{q} + C\dot{q} + Kq = F_{\text{ext}}(t) + r \tag{17}$$
where C and K are respectively the classical viscosity and stiffness matrices.

The typical dimension of the state vector can range between a few degrees of freedom and more than several hundred thousand, for example for mechanical or electrical systems.

Relations

As explained above, and according to Section 1 notation, some relations between local, (y,λ), and global variables (x,r), have to be set to described interactions between systems. Their general form is:
$$y = h(x,t,\ldots), \quad r = g(\lambda,t,\ldots) \tag{18}$$
Any other relations are derived from this one. Possible cases are:

- Linear time-invariant:
$$y = Cx + Fu + D\lambda + e, \quad r = B\lambda + a \tag{19}$$
- Lagrangian:
$$\dot{y} = H(q,t)\dot{q}, \quad r = H^t(q,t)\lambda \tag{20}$$
- Lagrangian Linear:
$$y = H\dot{q} + b, \quad r = H^t\lambda \tag{21}$$

Note that, $H(q,t)$ can optionally depend on u or λ.

Nonsmooth laws

- Complementarity condition or unilateral contact:
$$0 \le y \perp \lambda \ge 0 \tag{22}$$
- Newton impact:
$$\text{if } y(t) = 0, \quad 0 \le \dot{y}(t^+) + e\dot{y}(t^-) \perp \lambda \ge 0 \tag{23}$$
- Relay:
$$\begin{cases} \dot{y} = 0 & |\lambda| \le 1 \\ \dot{y} \ne 0 & \lambda = \text{sgn}(y) \end{cases} \tag{24}$$
- Unilateral contact and Coulomb's Friction, with $y = [y_n, y_t]^T, \lambda = [\lambda_n, \lambda_t]^T$:
$$\text{if } y_n = 0, \begin{cases} 0 \le \dot{y}_n \perp \lambda_n \ge 0 \\ \dot{y}_t = 0, \quad \|\lambda_t\| \le \mu\lambda_n \\ \dot{y}_t \ne 0, \quad \lambda_t \le -\mu\lambda_n \,\text{sgn}(\dot{y}_t) \end{cases} \tag{25}$$
- Piece-wise linear relations with fill in graphs as depicted on the figure 4.

The complete NSDS

At the end, the complete NSDS object contains a list of dynamical systems and of interactions between them. An object `Topology` has been introduced, as a field of NSDS that "knows" all interactions and their links. That especially allows us to define the relative degrees between output and input of the different coupled systems.

2.3 Simulation of NSDS behaviour

The various possible strategies of simulation in Siconos are listed below:

- *Time-stepping* schemes.
- *Event-driven* schemes.
- Hybrid time integration methods between time-stepping and event-driven.
- Equilibrium analysis.
- Further ones dedicated to control and analysis (stability, bifurcations ...).

From a practical point of view, all those strategies are based on low level algorithms from Numerics, such as linear complementary problem, quadratic programming or non smooth Newton solvers.

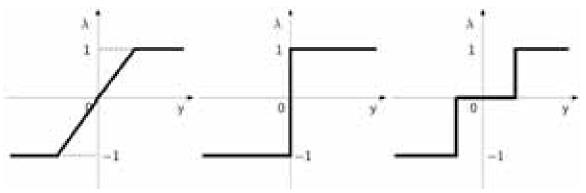


Figure 4: some multivalued piecewise linear laws: saturation, relay, relay with dead zone.

To conclude, general simulation strategy is summarized in the classes diagram of figure 3.b., with specific Siconos simulation objects, `OneStepIntegrator`, i.e. integrators for the dynamics without its non smooth part (Lsodar, Adams, Moreau ...) and `OneStepNSProblem`, which concerns the whole non smooth system.

3 Samples

In this part, specific examples of nonsmooth problems, solved with Siconos, are briefly presented: an electrical and two mechanical systems.

3.1 A first order system: electrical oscillator with 4 diodes bridge full-wave rectifier

Description of the physical problem

In this sample, a LC oscillator initialized with a given voltage across the capacitor and a null current through the inductor provides energy to a load resistance through a full-wave rectifier that consists in a 4 ideal diodes bridge (see fig. 5). Both waves of the oscillating voltage across the LC are provided to the resistor with current flowing always in the same direction. The energy is dissipated in the resistor, resulting in a damped oscillation.

Formalizing the linear complementary problem

A nonsmooth law describes the behavior of each diode of the bridge as a complementarity condition between current and reverse voltage (variables (y,λ)). Depending on the diode position in the bridge, y stands for the reverse voltage across the diode or for the diode current. There is only one dynamical system, which is linear, the whole oscillator that interacts with itself, through a linear time invariant relation between the state variable x and the non smooth law variables (y,λ).

Using Kirchhoff current and voltage laws and branches constitutive equations, we obtain the following linear dynamic system (see figure 5 for notations):

$$\begin{pmatrix} \dot{v}_L \\ \dot{i}_L \end{pmatrix} = \begin{pmatrix} 0 & -1/C \\ 1/L & 0 \end{pmatrix} \begin{pmatrix} v_L \\ i_L \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ -1/C & 0 \\ 1/C & 0 \end{pmatrix}^T \begin{pmatrix} -v_{DR1} \\ -v_{DF2} \\ i_{DF1} \\ i_{DR2} \end{pmatrix} \quad (26)$$

that fits in the frame of (15) with $x = (v_L \quad i_L)^T$ and $\lambda = (-v_{DR1} \quad -v_{DR2} \quad i_{DF1} \quad i_{DR2})^T$. Thus, linear relations are

$$r = B\lambda = \begin{pmatrix} 0 & 0 & -1/C & 1/C \\ 0 & 0 & 0 & 0 \end{pmatrix} (-v_{DR1} \quad -v_{DR2} \quad i_{DF1} \quad i_{DR2})^T$$

and

$$y = \begin{pmatrix} i_{DR1} \\ i_{DF2} \\ -v_{DF1} \\ -v_{DR2} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} v_L \\ i_L \end{pmatrix} + \begin{pmatrix} 1/R & 1/R & -1 & 0 \\ 1/R & 1/R & 0 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} -v_{DR1} \\ -v_{DF2} \\ i_{DF1} \\ i_{DR2} \end{pmatrix}$$

which corresponds to linear time-invariant relations of the form (19). Finally, complementarity conditions result from the ideal diode characteristic, and provide the required nonsmooth law:

$$0 \le -v_{DR1} \perp i_{DR1} \ge 0, \qquad 0 \le -v_{DF2} \perp i_{DF2} \ge 0$$
$$0 \le i_{DF1} \perp -v_{DF2} \ge 0, \qquad 0 \le i_{DR2} \perp -v_{DR2} \ge 0$$

Simulation and comparison with numerical results coming from SPICE models and algorithms

Siconos results are compared with those obtained using the SMASH simulator from Dolphin, with a smooth model of the diode as given by Shockley's law, a classical one step solver (Newton-Raphson) and the trapezoidal integrator.

Figure (6).a depicts the static $I(V)$ characteristic of two diodes with default SPICE parameters and two values for the emission coefficient N: 1.0 (standard

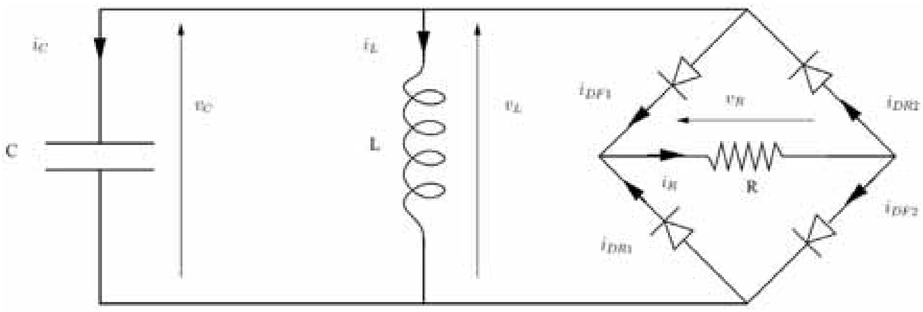


Figure 5: Electrical oscillator with 4 diodes bridge full-wave rectifier

diode) and 0.25 (stiff diode). The stiff diode is close to an ideal one with a threshold of 0.2 V.

Figure (6).b displays a comparison of the SMASH and SICONOS results with a trapezoidal integration ($\theta = 0.5$) and a fixed time step of 1 µs. A stiff diode model was used in SMASH simulations. One can notice that the results from both simulators are very close. The computational effort is much larger with Spice, which require a lot of Newton iterations to solve this very stiff and smooth problem, compared with a simple LCP solving.

3.2 A Lagrangian nonlinear system: simulation of a Robotic Arm

As a second example, we consider now the Mitsubishi PA-10 Robot, a seven-degree of freedom anthropomorphic robot, presented on figure 7. In Siconos, this robotic arm is modeled using a Lagrangian (nonlinear) dynamical system, that interacts with the ground. The arm falls down due to its own weight and bounces on the ground. Moreover, articular stops have been included to limit angular rotations. The degrees of freedom are the rotation angles, represented by the vector q. Neither external forces nor control terms are present. Finally, the robot dynamics

(a)

(b)

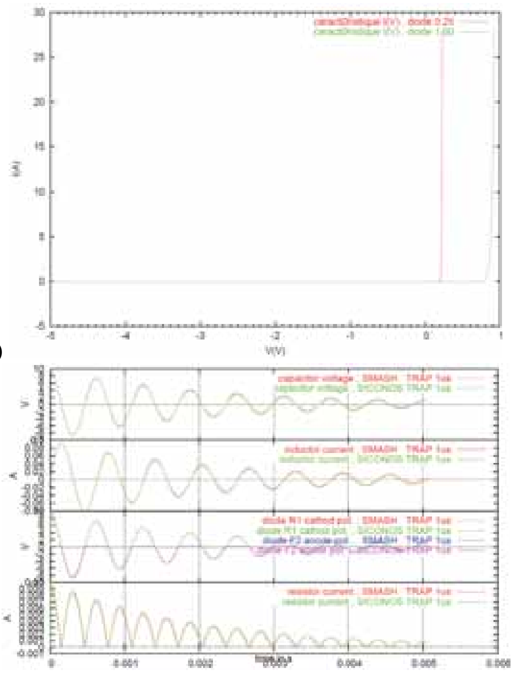


Figure 6: (a) Diodes characteristics from SPICE model with $N = 0.25$ and $N = 1$
(b) SMASH and SICONOS simulation results with trapezoidal integration, 1 µs time step.

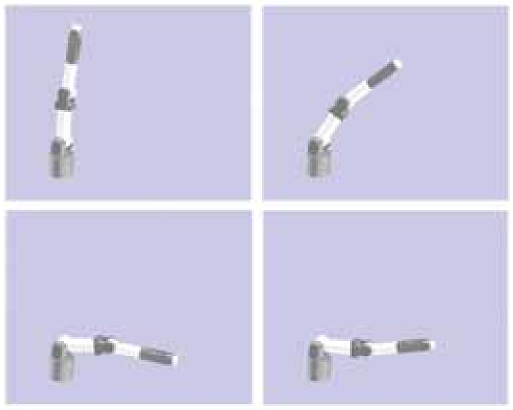


Figure 7: robotic arm fall-down: (a) initial position, (b) before contact with ground, (c) contact, (d) post-contact.

falls into the framework of Lagrangian dynamical systems as in (16). As a relation, we set that distance between arm and ground must remain positive, which means that contact can occurs but without penetration. The contact is supposed to be frictionless with a restitution coefficient denoted as e. This leads to nonlinear links between angular position (i.e. components of q), that can be written as:

$$\dot{y} = h(q)\,\dot{q}, \quad r = g(q), \quad \lambda = \nabla^t_{\ q} h(q)\lambda \qquad (27)$$

A Newton-Impact non smooth law complete this set of equations. with a restitution coefficient $e = 0.9$ for contact with the ground and $e = 0$ for angular stops.

The results of Siconos simulation, using Moreau time-stepping, Newton algorithm and a non smooth quadratic programming solver, are presented on figures 7 and 8. We denote A and B respectively the first (the one clamped on the ground) and second parts of the arm; θ_1 is the angle between A and the vertical position and θ_2 the angle between A and B. First, θ_1 and θ_2 increase, until the last one reaches its maximum angular position. Then θ_1, goes on increasing and θ_2 remains constant until the extremity of the arm touches the ground and bounces, here with a restitution coefficient e equal to 0.9. Finally A touches the ground and bounces also, together with B, until the complete arm lies on the ground. Note on 8.c and d that angular velocity cancels when angular stops are reached and change their signs when contact is established.

3.3 Beads column

We consider here a column of 1000 spherical beads, in contact or not, falling down to the ground. Dy-

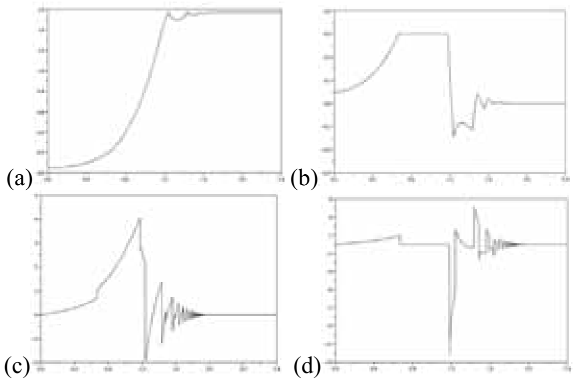


Figure 8: (a) $\theta_1(t)$, (b) $\theta_2(t)$, (c) $\dot{\theta}_1(t)$, (d) $\dot{\theta}_2(t)$

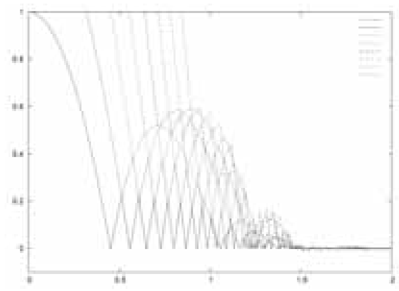


Figure 9: Vertical displacement of beads according to time

namical systems are Lagrangian ones and relations of type Lagrangian Linear, this with a Newton impact law. Figure 9 displays vertical displacements of the 8 lowest beads according to time. The interest of this example lies in the important number of degrees of freedom (i.e. size of vector q) and of relations (size of y and λ) equal to 1000.

Acknowledgements
This work is supported by the European Community through the Information Society Technologies thematic programme under the project SICONOS (IST-2001-37172).

References

[1] B. Brogliato. *Nonsmooth Mechanics: Models, Dynamics and Control*. Communications and Control Engineering. Springer-Verlag, second edition, 1999.

[2] F. H. Clarke. *Optimization and Nonsmooth analysis*. Wiley, New York, 1983.

[3] R. W. Cottle, J. Pang, R. E. Stone. *The linear complementarity problem*. Academic Press, Inc., Boston, MA, 1992.

[4] H. Elmqvist, S.E. Mattsson, M. Otter. *Object-oriented and hybrid modeling in Modelica*. Journal Européen des systèmes automatisés, 35(4):395 – 404, 2001.

[5] W. P. M. H. Heemels, J. M. Schumacher, S. Weiland. *Linear complementarity systems*. SIAM journal of Applied Mathematics, 60(4):1234–1269, 2000.

[6] T. A. Henzinger. *The theory of hybrid automata*. In M. K. Inan and R. P. Kurshan, editors, *Verification of Digital and Hybrid Systems*, volume 170 of NATO ASI Series F: Computer and Systems Sciences, pages 265-292. Springer Verlag, 2000.

[7] M. D. P. Monteiro Marques. *Differential Inclusions in Nonsmooth Mechanical Problems: Shocks and Dry Friction*. Birkhauser, Verlag, 1993.

[8] J. J. Moreau. *Unilateral contact and dry friction in finite freedom dynamics*. In J. J. Moreau and P. D. Panagiotopoulos, editors, *Nonsmooth mechanics and applications*, number 302 in CISM, Courses and Lectures, pages 1–82. Springer Verlag, 1988.

[9] J. S. Pang, R. Chandrasekaran. *Linear complementarity problems solvable by a polynomially bouded pivoting algorithms*. Math. Prog. Study, 25:13–27, 1985

[10] S. M. Robinson. *Generalized equations and their solutions. I. Basic theory*. Mathematical programming study, 10:128–141, 1979.

[11] M. Schatzman. *A class of nonlinear differential equations of second order in time*. Non linear Analysis, 2(3):355–373, 1978.

[12] van der Schaft, J. M. Schumacher. *An Introduction to Hybrid Dynamical Systems*. Springer Verlag London, 2000.

Corresponding author: Vincent Acary,
INRIA Rhône-Alpes – Bipop project
Zirst, 655 avenue de l'Europe
Montbonnot, 38334 Saint Ismier Cedex, France
vincent.acary@inrialpes.fr

Accepted MATHMOD 2006, January 2006
Received: June 12, 2006
Revised: January 20, 2007
Accepted: February 10, 2007

SNE 17/3-4, December 2007

Yo-yo Simulation on the Base of Analytical Treatment

Rüdiger Hohmann, University of Magdeburg, Germany, *hohmann@isg.cs.uni-magdeburg.de*
Lutz Sperling, University of Magdeburg, Germany, *sperling.md@arcor.de*

The benchmark refers to a competition in the yo-yo play during the 4th Vienna Mathmod conference. The ideal case is a cyclic motion of the yo-yo body and the player's hand which can ,e.g., be created by control. Under simplifying prerequisites one can also receive an analytical solution. A semi-elastic impact and a Dirac acceleration impulse at the lower reversal point as well as a vertical yo-yo motion are such physical prerequisites. Opposite to a finite impulse the Dirac impulse is favourable for an analytical treatment. Periodicity conditions define algebraic equations for the determination of parameters. The simulation offers the opportunity to confirm these results. State events are an adequate construct for the modelling of the reversal points. The comparison of two restitution coefficients of various size shows the high parameter sensitiveness of the solutions. The delta function, the Heavyside function and the Macauley bracket symbol permit to give equations of motions and their integrals for the whole period of motion.

Introduction

The aim of the present paper is to contribute to the benchmark of the computer simulation of a yo-yo.

During the basic yo-yo play the player tries to maintain a stable alternate motion of the yo-yo downwards and upwards as long as possible. Thus, the ideal case is a periodic motion of the yo-yo body and the player's hand.

Hence, on the basis of a yo-yo model that is as simple as possible, a practical periodic motion is analytically determined and simulated on the computer.

In some contributions the inclusion of the delta function into continuous models has been investigated [2, 3]. Opposite a finite impulse the Dirac impulse is also favourable for an analytical treatment. The acceleration impact of the hand at the reversal point is exemplary for this.

1 Model and equation of motion

The model consists of a rigid yo-yo body and a rigid massless string. The axle, into which the string is wound, has the constant radius r. This means that we disregard the change of r, because of a changing part of string on the axle.

Figure 1 shows the yo-yo model in a sinistral position. Its dexter position corresponds to Figure 1 in a side-inverted form.

Only the axle of the axially symmetric yo-yo body is illustrated in the figure. The complete yo-yo body has the mass m, the mass centre S, and the moment of inertia J_S with respect to the axis through S. The acceleration of gravity is g.

O is a space-fixed point, and A is the point where the hand of the player holds the string. The function $u(t)$ describes the vertical motion of point A.

The coordinate $s(t)$ describes the relative motion of point S in vertical direction opposite to point A. The current value of the coordinate s is identical with the length of the unwinded string. The total length of the string is l. The force of the string is F_S.

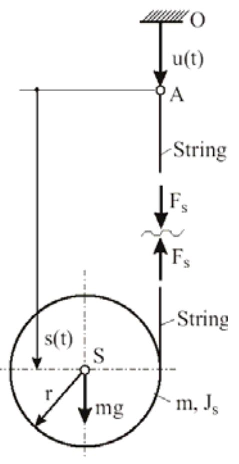


Figure 1. Yo-yo model

The string is assumed to be always taut and vertical. If necessary, the validity of this assumption shall be checked.

When the whole string is unwinded at the sinistral body position, a reversion of motion takes place changing the position to the right side and vice versa.

During the rolling motion no horizontal forces act on the yo-yo body, provided that point A is not moved horizontally. Then it is guaranteed that point S only moves vertically and that the string maintains its vertical direction.

The principle of linear momentum in the vertical direction yields to

$$m\ddot{s} + F_S = m(g - \ddot{u}) \qquad (1)$$

The principle of angular momentum with respect to the axis through mass centre S results in

$$F_S = \frac{J_S}{r^2}\ddot{s} \tag{2}$$

By substituting this expression for F_S in equation (1) we obtain the equation of motion:

$$\ddot{s} = k\left(g - \ddot{u}\right) \tag{3}$$

with the ratio of moments of inertia

$$k = \frac{mr^2}{J_S + mr^2} \tag{4}$$

The formulae (1) to (4) apply for the sinistral as well as for the dexter position of the yo-yo body. According to equations (2) and (3) the tautness of the string ($F_S \geq 0$) is guaranteed under the condition $\ddot{u} \leq g$.

When generating the yo-yo model, special attention must be paid to the reversion of motion in case of a completely unwinded string, and the energy loss through dissipation, which has to be compensated by the motion of the player's hand during the whole period of motion. For simplicity, both problems are treated similarly to Newton's elementary theory of impact. Thus, the rolling motion is assumed to proceed without any energy loss, and we abstain from the detailed modelling of the temporary segment of motion when the string is completely unwinded. A possible sideward motion of point S during this phase shall immediately be compensated by an equivalent hand motion, such that the string is vertical again when the following rolling motion begins.

For the velocity $v_{pre} = \dot{s}\left(s = l\right)$ of the yo-yo body with respect to the hand motion achieved immediately before the reversion of motion, the velocity after reversion is assumed to be $v_{post} = -ev_{pre}$. Coefficient e corresponds to Newton's impact coefficient and may be identified by a simple experiment. For that purpose, we let the yo-yo body freely role downwards from the position $s = 0$ and observe to which maximum height ηl (i.e. $s = l(1-\eta)$), $0 < \eta < 1$ the role moves up again after the reversion. Then, corresponding to the impact theory, the coefficient e is identified by $e = \sqrt{\eta}$.

2 Analytical derivation of a periodic solution

In the following, a solution shall be derived that is periodic both in $s(t)$ and $u(t)$. For this purpose, the function $u(t)$ for the hand motion is defined in a simple analytical form with free parameters, which shall be identified by means of certain periodicity conditions.

The motion is assumed to start at the moment $t = 0$ with positions $u = 0$ and $s = 0$. In the first segment of motion $0 \leq t < t_r$ the hand moves downwards with constant velocity $\dot{u}(t) = v_d = const.$, $v_d > 0$ without acceleration, whereas t_r is the moment of the lower reversion of motion. Immediately, i. e. a differential duration before $t = t_r$, the player may apply an upwards directed Dirac impulse of acceleration $-v_s\delta(t - t_r)$. Thus, the velocity of the hand is abruptly changed to the new negative value $v_d - v_s < 0$. $\delta(t)$ is the Dirac delta function.

During the second motion segment $t_r < t \leq T$ with T as period the hand moves with the constant downwards directed acceleration $a = const. > 0$ until it has reached again the primary velocity v_d. Thus, the hand motion has a reversion before the period ends.

$$\ddot{u} = \begin{cases} 0 & 0 \leq t < t_r & (5) \\ -v_s\,\delta(t - t_r) & t = t_r & (6) \\ a & t_r < t \leq T & (7) \end{cases}$$

By integrating the acceleration and by considering the initial conditions we get the corresponding expressions for velocity and hand position. For the first segment of motion $0 \leq t < t_r$ this results in $\dot{u}(t) = v_d > 0$, $u(t) = v_d t$. For the moment immediately after the acceleration delta impulse we get $\dot{u}(t_r) = v_d - v_s < 0$, $u(t_r) = v_d t_r$. The results for the second segment $t_r < t \leq T$ are

$$\dot{u}(t) = a\left(t - t_r\right) + v_d - v_s \tag{8}$$

$$u(t) = \frac{1}{2}a\left(t - t_r\right)^2 + \left(v_d - v_s\right)\left(t - t_r\right) + v_d t_r \tag{9}$$

For the point $t = T$ this yields to the periodicity conditions

$$a\left(T - t_r\right) + v_d - v_s = v_d \tag{10}$$

$$\frac{1}{2}a\left(T - t_r\right)^2 + \left(v_d - v_s\right)\left(T - t_r\right) + v_d t_r = 0 \tag{11}$$

Now, the motion of the yo-yo body is considered. At the beginning of the motion its relative motion with respect to the hand has a reversion of the direction $\dot{s}(t = 0) = 0$. Thus, the integration of the equation of motion (3) under consideration of equation (5) for the first segment of motion $0 \leq t < t_1$ yields to

$$\dot{s}(t) = kgt \tag{12}$$

$$s(t) = \frac{1}{2}kgt^2 \tag{13}$$

According to equation (3) not only the hand but also the yo-yo body obtains an acceleration Dirac impulse immediately before the moment $t = t_r$. Therefore, at the moment $t = t_r$ it is already subject to energy loss through the abrupt reversion of motion: $\dot{s}(t_r) = -e\left[\dot{s}(t_r - 0) + kv_s\right]$. Under consideration of equation (12), we obtain

$$\dot{s}(t_r) = -ek(gt_r + v_s) \tag{14}$$

From equation (13) under consideration of condition $s(t = t_r) = l$, we get

$$t_r = \sqrt{\frac{2l}{kg}} \tag{15}$$

Under consideration of equation (14) the second segment of motion $t_r < t \le T$ results in

$$\dot{s}(t) = k(g-a)(t-t_r) - ek(gt_r + v_s) \tag{16}$$

$$\begin{aligned} s(t) = &\frac{1}{2}k(g-a)(t-t_r)^2 \\ &-ek(gt_r + v_s)(t-t_r) + l \end{aligned} \tag{17}$$

For $t = T$ we obtain the conditions of periodicity

$$k(g-a)(T-t_r) - ek(gt_r + v_s) = 0 \tag{18}$$

$$\begin{aligned} &\frac{1}{2}k(g-a)(T-t_r)^2 \\ &-ek(gt_r + v_s)(T-t_r) + l = 0 \end{aligned} \tag{19}$$

Now, the periodicity conditions (10), (11), (18) and (19) are 4 algebraic equations for the parameters v_d, v_s, a of the hand motion and for the whole period of motion T.

With the abbreviation $\tau = T - t_r$ equation (10) results in

$$a\,\tau = v_s \tag{20}$$

Hence, for the unknown parameters v_d, v_s, τ the equations

$$(2v_d - v_s)\tau + 2v_d t_r = 0 \tag{21}$$

$$g\tau - v_s - e(gt_r + v_s) = 0 \tag{22}$$

$$\begin{aligned} &g(\tau^2 + t_r^2) - v_s\,\tau \\ &-2e(gt_r + v_s)\,\tau = 0 \end{aligned} \tag{23}$$

remain with t_r according to equation (15).

In spite of the very simple construction of the hand acceleration this is a nonlinear system of algebraic equations, which is, however, reducible to a quadratic equation.

Substituting v_s by the expression

$$v_s = \frac{\tau^2 - t_r^2}{\tau}g \tag{24}$$

following from equations (2.18) and (2.19), equation (2.18) can be transformed into the quadratic equation

$$\tau^2 + t_r\tau - t_r^2\left(1 + \frac{1}{e}\right) = 0 \tag{25}$$

The positive solution is

$$\tau = \frac{1}{2}(d-1)t_r \tag{26}$$

with the abbreviation

$$d = \sqrt{5 + \frac{4}{e}} \tag{27}$$

The period T is

$$T = \frac{1}{2}(d+1)t_r \tag{28}$$

From equations (21) and (24) substituting τ according to equation (26) we get

$$v_d = (\tau - t_r)\frac{g}{2} = \frac{1}{4}(d-3)gt_r \tag{29}$$

and

$$v_s = \frac{1}{2}\left(\frac{d-e}{1+e} - 1\right)gt_r \tag{30}$$

At last, for the downwards directed acceleration in the second segment of motion, equation (20) yields to

$$a = \frac{v_s}{\tau} = \frac{1}{d-1}\left(\frac{d-e}{1+e} - 1\right)g \tag{31}$$

For every value $e \ne 0$ ($e < 1$) this expression for *a* is smaller than the acceleration of gravity *g*, and thus the string remains taut. In the borderline case $e = 0$ a periodic yo-yo motion is impossible, and after the initial downward motion the string remains always completely unwinded.

In the special case of the conservative system, i. e. for $e = 1$, we have

$$\tau = t_r,\ T = 2t_r,\ v_d = 0,\ v_s = 0,\ a = 0 \tag{32}$$

which means that the hand is always at rest and the yo-yo body rolls up and down ad infinitum without energy loss throughout the whole length of the string.

3 Simulation by use of state events

The continuous simulation offers the possibility to confirm results of models that where determined analytically from differential equations by numeric simulation. This way parameter studies can be carried out comfortably and graphic representations of the solution functions can be produced as well. As a rule, nonlinearities in the equations of motions do not form any problem.

The use of the delta function, the Heavyside function and the Macauley bracket symbol permits to give expressions of the equations of motions and their integrals for the whole period of motion. The acceleration Dirac impulse is described by the delta function $\delta(t)$, while the discontinuous Heavyside function $H(t)$ serves for the changing of a.

$$H(t-t_r) = \int_0^t \delta(t-t_r)\mathrm{d}t = \begin{cases} 0 & \text{for} \quad t < t_r \\ 1 & \text{for} \quad t \geq t_r \end{cases} \tag{33}$$

The integrals, piecewise defined polynomials up to the order $n=2$, can be represented by the Macauley bracket symbol [1] (in German called Föppl symbol). The Macauley symbol of 0th order ($n=0$) is identical with the Heavyside function $H(t-t_r)$:

$$\langle t-t_r \rangle^n = \begin{cases} 0 & \text{for} \quad t < t_r \\ (t-t_r)^n & \text{for} \quad t \geq t_r \end{cases} \tag{34}$$

$$\int_0^t \langle t-t_r \rangle^n \mathrm{d}t = \frac{1}{(n+1)} \langle t-t_r \rangle^{(n+1)} \tag{35}$$

In the differentiation and the integration the bracket symbol indicated by pointed brackets is treated just as round brackets. Equations of motions for hand and yo-yo are:

$$\ddot{u} = a\langle t-t_r \rangle^0 - v_s \delta(t-(t_r-0)) \tag{36}$$

$$\begin{aligned} \ddot{s} = kg + kv_s \delta(t-(t_r-0)) - ka\langle t-t_r \rangle^0 \\ -\left[(1+e)kgt_r + (1+e)kv_s\right]\delta(t-t_r) \end{aligned} \tag{37}$$

The second term of (37) refers to the Dirac impulse of the hand acceleration at $t = t_r - 0$, the terms in the square brackets refer to the semi-elastic impact at $t = t_r$, consisting of compression phase and restitution with coefficient e.

The investigations were carried out by using the simulation system ACSL (Advanced Continuous Simulation Language). A block-equation oriented form is characteristic. Besides algebraic equations, blocks are available for nonlinearities and especially for the integration. The model notation starts with the constant parameter values, followed by the calculation of derived quantities. Here, the times are expressed as t_r and τ and the speeds as v_d and v_s, besides the relationship of the moments of inertia k and the hand acceleration a. Variables, represented in italics in the analytical context, appear in the standard form in the model notation.

The used parameter values are $l = 1\,\mathrm{m}$, $m = 0.03\,\mathrm{kg}$, $r = 0.008\,\mathrm{m}$, $J_s = 10^{-5}\,\mathrm{kg\,m^2}$. The radius of 5 mm provided in the Benchmark shall enlarge to 8 mm on average by the rolled up string.

The Dynamic section contains the model equations in the Derivative section and two Discrete sections with instructions at the lower and upper reversal point. With exception of the lower reversal point the following expressions of the model notation apply to the accelerations `s2d` and `u2d` while the effect caused by the impact, is represented by the Discrete section `REVERSAL`.

```
s2d = k*(g - a)          ! yo-yo acceleration
u2d = a                  ! hand acceleration
w = u + s                ! absolute yo-yo position
wd = ud + sd             ! absolute yo-yo velocity
SCHEDULE  Reversal .XP. (s-l)
SCHEDULE  New .XP. sd
```

The different simulations are based on the initial values $a(t=0) = 0$, s0 = sd0 = u0 = 0, ud0 = v_d.

Both state events at the reversal points are activated by zero crossings from negative into the positive of time functions `(s-l)(t)` and `sd(t)` after iterations. Because the simulation system integrates up to the event time and restarts from there with changed parameters, a numerically critical integration over discontinuities does not take place this way.

The Discrete section `REVERSAL` sets the hand acceleration on `a`$(t = t_r)$`=`$a > 0$ in the lower reversal point, the hand velocity `ud=`$v_d - v_s$, and the relative velocity after the impact `sd = -`e`*(sd+`k`*`v_s`)`.

A final call `CALL ZZDERV(1)` actualizes the model equations with the new parameter values. Instructions `CALL LOGD(.TRUE.)` write additional data sets into the output file immediately before and after the impact, which is helpful for the graphic representation of discontinuities.

The second Discrete section `NEW` assigns the start value `a=0` to the hand acceleration at the upper reversal point again, and correct appeared numeric devia-

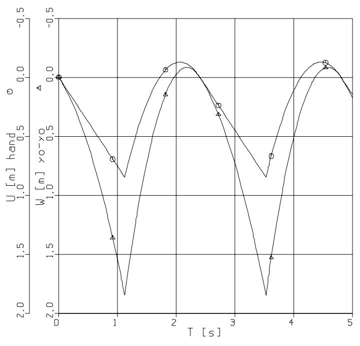


Figure 2. Hand and absolute yo-yo positions. Restitution coefficient $e = 0.7$

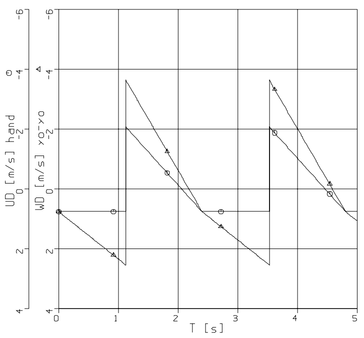


Figure 3. Hand and absolute yo-yo velocities. Restitution coefficient $e = 0.7$

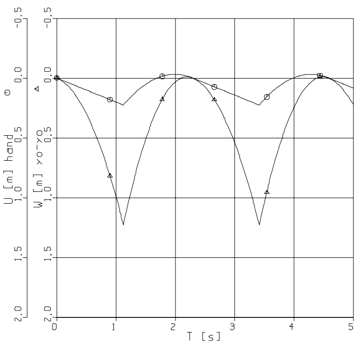


Figure 4. Hand and absolute yo-yo positions. Restitution coefficient $e = 0.9$

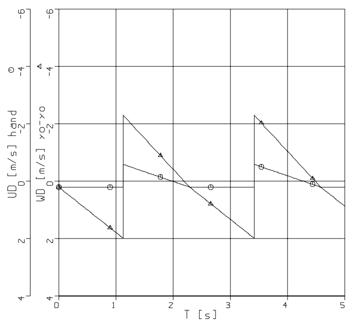


Figure 5. Hand and absolute yo-yo velocities. Restitution coefficient $e = 0.9$

tions by the initial conditions `u=0`, `s=sd=0`, and `ud=` v_d.

The comparison of two restitution coefficients of various size shows the high parameter sensitiveness of the solution functions.

In figures 2 to 5, resulting from simulations, the positions and velocities of hand and yo-yo are compared. At first the chosen restitution coefficient is $e = 0.7$ which increases to $e = 0.9$. The corresponding values of η are $\eta = 0.49$ and $\eta = 0.81$. These are the heights in the dimension [m] reached again at immovable hand. The more the impact at the reversal point approaches the elastic impact, the less is the necessary motion of the hand for the compensation of energy losses.

4 Conclusion

In this paper an analytical solution of the benchmark task of a yo-yo has been determined and confirmed by numeric simulation. Physical prerequisites are a vertical yo-yo motion, a semi-elastic impact and a Dirac acceleration impulse at the lower reversal point. Opposite to a finite impulse the Dirac impulse is favourable for an analytical treatment, since only beginning and final state are considered, but not the process during the impulse itself.

Periodicity conditions define the ideal case of a periodic motion of the yo-yo body and the player's hand. It can also be obtained by control. A haptic interface, with which the player moves a yo-yo model by means of real gestures is also conceivable. Fig. 2 and Fig. 4 show that the hand motions can reduce if the restitution coefficient approaches $e = 1$, the value of an elastic impact.

References

[1] D. Gross, W. Hauger, W. Schnell, J. Schröder: *Technische Mechanik 1, Statik, 8. Auflage.* Berlin Heidelberg: Springer-Verlag, 2004.

[2] R. Hohmann: *Delta- and Theta function in Continuous Models.* The Proceedings of the 1994 Summer Computer Simulation Conference, pp. 775-779, San Diego, California, USA, 1994.

[3] R. Hohmann: *Methoden und Modelle der Kontinuierlichen Simulation.* Habilitationsschrift, Shaker Verlag, Aachen 1999.

Corresponding author: Rüdiger Hohmann,
University of Magdeburg, Germany
hohmann@isg.cs.uni-magdeburg.de

Received: September 12, June 2007
Revised: October 15, 2007
Accepted: October 20, 2007

SNE 17/3-4, December 2007

Hybrid Performance Modeling and Prediction of Parallel and Distributed Computing System

Sabri Pllana and Siegfried Benkner, University of Vienna, Austria, *{pllana,sigi}@par.univie.ac.at*
Felix Breitenecker, Vienna University of Technology, Austria, *felix.breitenecker@tuwien.ac.at*

Performance is a key feature of large-scale computing systems. However, the achieved performance when a certain program is executed is significantly lower than the maximal theoretical performance of the large-scale computing system. The model-based performance evaluation may be used to support the performance-oriented program development for large-scale computing systems. In this paper we present a hybrid approach for performance modeling and prediction of parallel and distributed computing systems, which combines mathematical modeling and discrete-event simulation. We use mathematical modeling to develop parameterized performance models for components of the system. Thereafter, we use discrete-event simulation to describe the structure of system and the interaction among its components. As a result, we obtain a high-level performance model, which combines the evaluation speed of mathematical models with the structure awareness and fidelity of the simulation model. We evaluate our approach with a real-world material science program that comprises more than 15,000 lines of code.

Introduction

The solution of resource-demanding scientific and engineering computational problems involves the execution of programs on parallel and distributed computing systems, which commonly consist of multiple computational nodes, in order to solve large problems or to reduce the time to solution for a single problem. However, there is a widening gap between the maximal theoretical performance and the achieved performance when a certain program is executed on a parallel and distributed computing system. This gap may be reduced by tuning the performance of a program for a specific computing system. Commonly, the programmer develops multiple versions of the program following various parallelization strategies. Thereafter, the programmer assesses the performance of each program version, and selects the program version that achieves the highest performance. The code-based performance tuning of a program is a time-consuming and error-prone process that involves many cycles of code editing, compilation, execution, and performance analysis. This problem may be alleviated by using the model-based performance evaluation.

In this paper we present a methodology and the corresponding tool-support for performance modeling and prediction of parallel and distributed computing systems, which may be used in the process of performance-oriented program development for providing performance prediction results starting from the early program development stages. Based on the performance model, the performance can be predicted and design decisions can be influenced without time-consuming modifications of large parts of an implemented program.

We propose a hybrid approach for performance modeling and prediction of parallel and distributed computing systems, which combines mathematical modeling and discrete-event simulation. Our aim is to combine the evaluation speed of mathematical models with the structure awareness and fidelity of the simulation model. For the purpose of evaluation of our approach we have developed a performance modeling and prediction system called Performance Prophet. We demonstrate the usefulness of Performance Prophet by modeling and simulating a real-world material science program that comprises more than 15,000 lines of code. In our case study, the model evaluation with Performance Prophet on a single processor workstation is several thousand times faster than the execution time of the real program on our cluster.

The rest of this chapter is organized as follows. Section 1 describes performance-oriented programming models and languages, and the issues related to the performance-oriented program development. Our approach for hybrid performance modeling and prediction of parallel and distributed computing systems is described in Section 2. We evaluate our approach in Section 3. Finally, Section 4 presents our conclusions and describes the future work.

1 Performance-oriented parallel and distributed programs

Commonly performance-oriented parallel and distributed programs are developed using imperative programming languages, such as FORTRAN or C, based on a suitable programming model.

Commonly performance-oriented parallel and distributed programs are developed using imperative programming languages, such as FORTRAN or C, based on a suitable programming model.

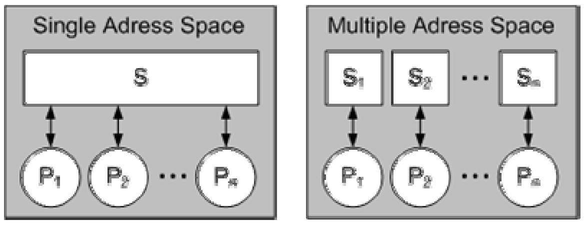


Figure 1. Programming models. (a) Shared memory, (b) message passing

Figure 1 depicts the commonly used parallel and distributed programming models. A *programming model* describes the relationship between processing elements and the address space. An *address space* is a collection of uniquely identified memory locations. Basically, the address space defines the scope of memory locations that can be addressed by the processing element.

Figure 1(a) depicts the shared memory programming model. All processing elements P_i, $1 \le i \le n$, share a single address space. This means that all processing elements can directly read and write data in the shared memory. However, synchronization mechanisms are required to coordinate the access of processing elements to the shared memory. The advantage of shared memory model is the easy of programming. A drawback of shared memory model is the difficulty to exploit the data locality for performance optimization. An implementation of the shared memory programming model is Open Multi Processing (OpenMP) [1].

The message passing programming model is depicted in Figure 1(b). In the context of message passing model, the processing elements communicate by sending and receiving messages. Each processing element P_i has direct access to the data in its address space S_i, $1 \le i \le n$. However, a processing element P_i needs to exchange messages with P_j in order to obtain the data that resides in the address space S_j of processing element P_j, $i \ne j$. By controlling the data locality it is possible to achieve a more effective use of the memory hierarchy. The drawback is that the programmer has to deal with a fragmented memory model. From the programmer's point of view large data structures are fragmented across multiple address spaces. The programmer has to deal with the implementation of distribution of the data. While with the message passing model a high performance may be achieved, the development of message passing programs is a time-consuming and difficult task. Examples of implementation of the message passing programming model include Parallel Virtual Machine (PVM) [7] and Message Passing Interface (MPI) [16].

1.1 Performance-oriented program development

There is a widening gap between the maximal theoretical performance of a computing machine and the achieved performance when a certain program is executed [3][9][17]. This gap may be reduced by tuning the performance of a program for a specific computing machine. The procedure for improvement of the program performance involves multiple cycles of *code editing*, *compilation*, and *execution* (see Figure 2(a)). For real-world programs this procedure of code-based performance tuning is time-consuming, expensive and error-prone. Furthermore, its applicability is limited to the available computing machines.

Model based performance tuning involves the *model editing*, model *transformation* to a form that is suitable for evaluation, and model *evaluation* (see Figure 2(b)). In this manner, instead of making experiments with the real program code on the real computing machine, we are able to experiment with the model of the program and the model of the computing machine. Model-based performance tuning is efficient, inexpensive, and non-intrusive. Moreover, it is applicable not only to the available computing machines but also to those that are under development or being planned.

2 Hybrid performance modeling and prediction

Commonly for performance modeling of computing systems is used mathematical modeling or discrete event simulation. When applied separately, each of these approaches has severe limitations.

Mathematical models commonly represent the whole computing system as a symbolic expression that lacks the structural information [4]. An example of a mathematical performance model that models the program execution time is expressed as follows,

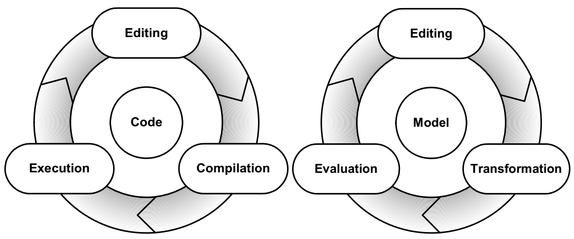
$$T_{ProgExec} = C_{Op} T_{Av}$$

where C_{Op} is the number of operations and T_{Av} is the average execution time of an operation. We may observe that there is no identifiable structural information in this model. The information such as the execution order of operations or the control flow is not contained in the model.

Detailed simulation models commonly are so slow that the assessment of real-world programs is impractical, or for the model evaluation are needed very large resources (processors and memory) that may not be available. For instance, RSIM is a simulator of CC-NUMA shared-memory machines [12]. RSIM comprises a detailed (that is a cycle-level) machine model that allows the analysis of the performance effects of architectural parameters. Therefore, it is suitable to evaluate various designs of CC-NUMA shared-memory machines. However, because the simulation of the program execution with RSIM is very slow (several thousands times slower than the program execution on the real machine), it is not suitable for evaluation of various designs of real-world programs.

Our aim is to combine the good features of both approaches. For instance, we would like to have the model evaluation efficiency of mathematical performance models and the structure awareness of simulation models. A model that combines *mathematical modeling* (MathMod) with *discrete-event simulation* (DES) is referred to as *hybrid model* [14].

Figure 3 shows that, considering the level of abstraction, the hybrid performance models of computing systems reside somewhere between MathMod models and DES models. An important feature of hybrid models is that they permit the system modeling at



(a) Code-based (b) Model-based

Figure 2. Performance tuning.

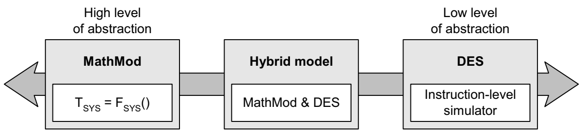


Figure 3. Hybrid performance models combine the features of MathMod models and DES models.

various levels of abstraction. The MathMod dominated hybrid models are at a higher level of abstraction and more efficient than the DES dominated hybrid models. On the other hand, the structure of system under study is modeled in more detail with the DES dominated hybrid models.

Figure 4 depicts the activity diagram of a hypothetical program. Activities $\{A_i \mid 1 \le i \le 4\}$ correspond to the code blocks of program. To each activity A_i is associated a parameterized *cost function* $F_{Ai}()$, which models the execution time of activity A_i. Functions $\{F_{Ai} \mid 1 \le i \le 4\}$ are obtained using the MathMod techniques. The structure of program, which includes activities and their order of execution, is described with DES.

Figure 5 depicts our hybrid approach for performance modeling of a parallel region. The parallel region executes activities *Computation1* and *Computation2* in parallel (see Figure 5(a)). Thread T_0 executes activity *Computation1*, whereas activity *Computation2* is executed by thread T_1 (see Figure 5(b)). The execution of activities *Computation1* and *Computation2* is modeled with MathMod. The synchronization of threads is modeled with DES.

Figure 6 depicts our hybrid approach for performance modeling of point-to-point interprocess communication. Process P_0, after performing some computation, sends a message to process P_1. Process P_1 receives the message. Computation is modeled with an activity

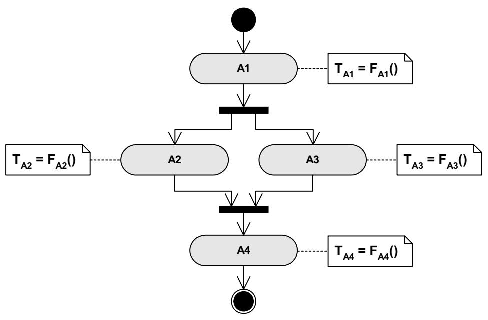


Figure 4. Hybrid performance model of a hypothetical program. The performance of activities is modeled with MathMod. The control flow is modeled with DES.

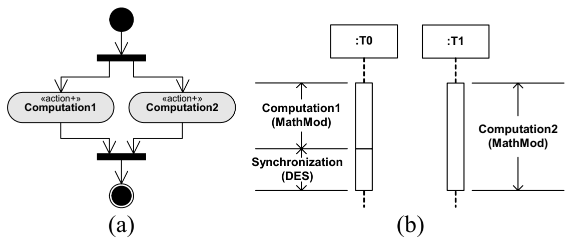


Figure 5. Hybrid performance modeling of a parallel region.

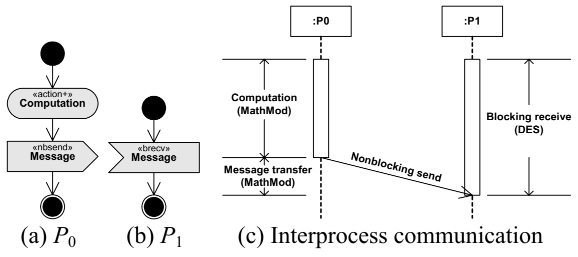


Figure 6. Hybrid performance modeling of point-to-point interprocess communication.

(see Figure 6(a)). The sending process P_0 uses a *non-blocking send* to send the message (see Figure 6(a)), whereas the receiving P_1 process uses a *blocking receive* to receive the message (see Figure 6(b)). *Computation* and the *message transfer* are modeled with MathMod, whereas waiting to receive the message is simulated with DES (see Figure 6(c)).

The model of parallel and distributed program (that is the workload model) is one of components of the computing system model. We apply the same methodology for building of the whole computing system model, which includes the machine model and the workload model. The behavior of the whole computing system is split-up into *action states* and *wait states*. Examples of action states include the execution of a code block such as a sequence of computational operations, or service time of a machine resource such as network subsystem. Wait states are used to model code blocks that involve multiple processing units such as parallel regions, or waiting for the availability of a machine resource such as a processor. While the duration of an action state is possible to determine in advance, in general it is not possible to determine in advance the duration of a wait state [8]. Therefore, we model the performance behavior of action states with MathMod techniques, whereas we simulate the behavior of wait states.

3 Evaluation

For the purpose of evaluation of our approach we have developed a performance modeling and prediction system called Performance Prophet [9].

3.1 Performance Prophet

Figure 7 depicts the architecture of Performance Prophet. The main components of Performance Prophet are *Teuta* and *Performance Estimator*. Teuta is a platform independent tool for graphical modeling of parallel and distributed programs [9][11]. The role of Performance Estimator, in the context of Performance Prophet, is to estimate the performance of a program on a computing machine.

Teuta comprises the following parts: *Model Checker*, *Model Traverser*, *Graphical User Interface* (GUI), and the components for *Performance Visualization* (see Figure 7). The GUI of Teuta is used for the development of performance model based on the Unified Modeling Language (UML) [6]. The Model Checker is used to verify whether the model conforms to the UML specification. The Model Traverser is used for generation of different model representations (XML and C++). The Performance Visualization components are used for visualization of the performance results.

Element MCF indicates the XML file, which is used for the model checking. The XML files that are used for the configuration of Teuta are indicated with the element CF.

The communication between Teuta and the Performance Estimator is done via elements PMP, SP and TF. Element PMP indicates the C++ representation of the program's performance model. PMP is generated by

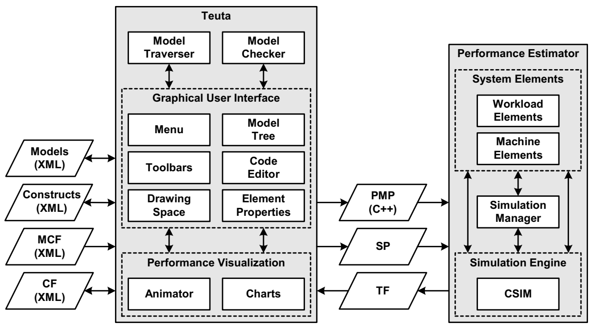
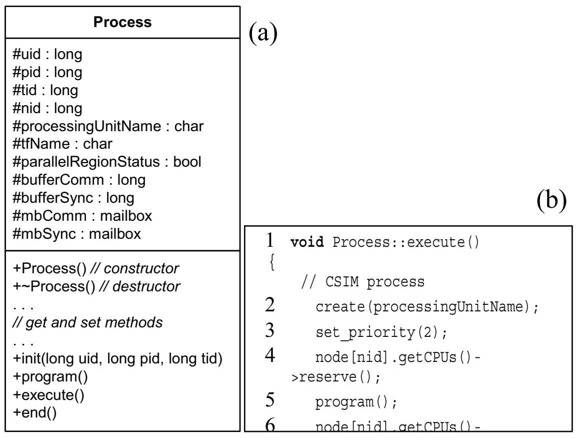


Figure 7. The architecture of Performance Prophet. Abbreviations: Model Checking File (MCF), Configuration File (CF), Performance Model of Program (PMP), System Parameters (SP), Trace File (TF).



```
Process
#uid : long
#pid : long
#tid : long
#nid : long
#processingUnitName : char
#tfName : char
#parallelRegionStatus : bool
#bufferComm : long
#bufferSync : long
#mbComm : mailbox
#mbSync : mailbox
+Process() // constructor
+~Process() // destructor
. . .
// get and set methods
. . .
+init(long uid, long pid, long tid)
+program()
+execute()
+end()
```

```
void Process::execute()
{
  // CSIM process
   create(processingUnitName);
   set_priority(2);
   node[nid].getCPUs()-
>reserve();
   program();
   node[nid].getCPUs()-
```

Figure 8. Class Process. (a) Structure; (b) Methode execute.

Teuta and serves as input information for the Performance Estimator. Element SP indicates a set of system parameters. The parameters of system include the number of computational nodes, the number of processors per node, the number of processes, and the number of threads. The Performance Estimator uses SP for building the model of system, whose performance is estimated. Element TF represents the trace file, which is generated by the Performance Estimator as a result of the performance evaluation. Teuta uses TF for the visualization of performance results.

The Performance Estimator comprises the following components: Simulation Manager, CSIM, Workload elements, and Machine elements (see Figure 7). The Simulation Manager builds the system model based on the user specification, starts and ends the simulation run, and stores the performance results. A set of Workload and Machine elements are provided for building of the system model. In what follows in this section we describe components of the Performance Estimator in more detail.

CSIM (Mesquite Software [5][15]) is a process-oriented general-purpose simulation library. CSIM supports the development of discrete-event simulation models, by using the standard programming languages C and C++. Because of the nature of compiled C and C++ programs and CSIM's dynamic memory allocation, the developed simulation models are compact and efficient. CSIM supports the process-oriented world view. The system is represented by a set of static components (that is CSIM *facilities*) and a set of dynamic components (that is CSIM *processes*) that use the static components. CSIM provides a set of abstractions (such as *processes* and *facilities*) for the model development, and many useful features (such as statistics collection or random variate generation) that are needed in a simulation study. CSIM processes operate in parallel in simulated time. Therefore, CSIM provides mechanisms for the synchronization of processes and for the interprocess communication. For the synchronization of CSIM processes are commonly used CSIM *events*. The communication among CSIM processes is accomplished via CSIM *mailboxes*.

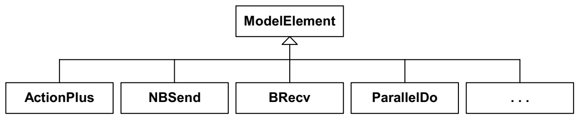


Figure 9. Performance modeling elements of program.

Based on CSIM we have developed a set of C++ classes that model basic program and machine components. Examples of these components include *Process*, *Send*, *Receive*, *ParallelDo*, and *Node*.

Figure 8 depicts the class Process, which we have developed to model processing units (that is processes or threads) of a computing system. The design of class Process permits the modeling of a large group of parallel and distributed scientific programs.

The structure of class Process is depicted in Figure 8(a). The unit ID (uid), process ID (pid) and thread ID (tid) are used to uniquely identify the processing unit during the simulation. The node ID (nid) indicates the computational node on which the processing unit is mapped. The attribute processingUnitName is mainly used to identify the processing unit in simulation reports. The performance evaluation results of processing unit are stored in the file that is specified in the attribute tfName. The attribute parallelRegionStatus indicates whether the processing unit is executing a parallel region (for instance, a code region enclosed within OpenMP directives PARALLEL and END PARALLEL [1]). The attributes bufferSync and mbSync serve for the synchronization among processing units. The communication among processing units is performed via attributes bufferComm and mbComm. We may observe that the CSIM type mailbox is used to define mbComm and mbSync. The methods of class Process for getting or setting values of attributes are straightforward, and therefore, they are not depicted in Figure 8(a). The methods init() and end() are invoked when the operation of process is initialized and completed respectively. The method program() models the perform-

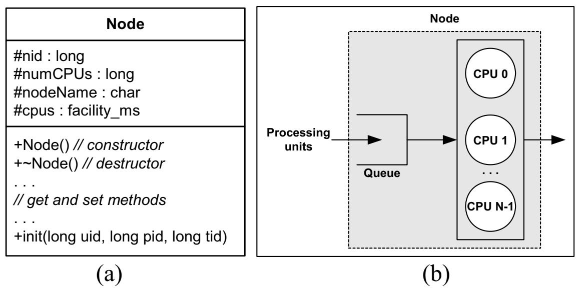


Figure 10. Class Node.

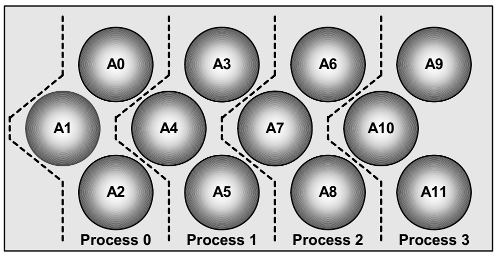


Figure 11. An instance of LAPW0 domain decomposition. Number of atoms (*NAT*) is 12; number of atoms per process (*PNAT*) is 3.

ance behavior of the program under study. Teuta generates automatically the code for the method program() based on the UML model that is specified by the user.

Figure 8(b) depicts the implementation of method execute() of class Process. In the line 3 is used the CSIM statement create() to define the method execute() as a CSIM process. The CSIM statement set_priority() sets the priority of the process (see line 4). Higher values of the priority mean higher priority of process execution. For instance, the process with priority 2 will execute before the process with priority 1 if the priority determines the order of execution. In the line 5 the process obtains the processor from the node. The statement in the line 6 invokes the method program(), which models the performance behavior of the program under study. In the line 7 the process releases the processor. Line 8 is used to notify the end of process execution.

Basically, the method program() of class Process specifies the execution flow of a collection of performance modeling elements. Each performance modeling element corresponds to a code block of the program, whose performance is modeled (see Figure 9). The execution of a performance modeling element models the performance behavior of a code block during the program execution.

Figure 9 depicts the hierarchy of classes of Performance Estimator that are used for construction of the method *program()*. On the top of hierarchy is the class *ModelElement*. The subclasses of class *ModelElement* correspond to various code blocks of parallel and distributed programs. A group of one or more program statements is referred to as a *code block*. Examples of subclasses of class *ModelElement* include: *ActionPlus*, *NBSend*, *BRecv*, and *ParallelDo*. Instances of these subclasses are used to represent the performance modeling elements in the method *program()* of class *Process*.

Figure 10 depicts the class *Node*, whose instances we use for modeling the computational nodes of computing machines. The structure of class *Node* is depicted in Figure 10(a). Attribute *nid* is used to uniquely identify instances of the class *Node*. The number of processors per node is specified with attribute *numCPUs*. Attribute *nodeName* is used to specify the name of the node. The CSIM type *facility_ms* is used to define processors of the node (that is *cpus*). The method *init()* of class *Node* is invoked when operation of the node is initialized.

Figure 10(b) depicts the structure of a node. A node comprises a set of processors $\{CPU_0, CPU_1, ..., CPU_{N-1}\}$ and a queue. Processing units (processes or threads) may use any of the available processors. If there is no processor available, then processing units wait in the queue. The default queue discipline is *first come first served*. Other queue disciplines, such as *round robin*, may be specified. If the *round robin* queue discipline is specified, then a processing unit uses a processor for the specified amount of time. Thereafter, the processing unit is preempted and the next processing unit that is waiting in the queue obtains the processor. Commonly, high performance programs are mapped on machines with sufficient hardware resources in the manner that processing units do not have to compete for processors. Nevertheless, the capability of simulation of situations when multiple processing units share one processor may be useful to reveal the performance drawbacks of such mappings.

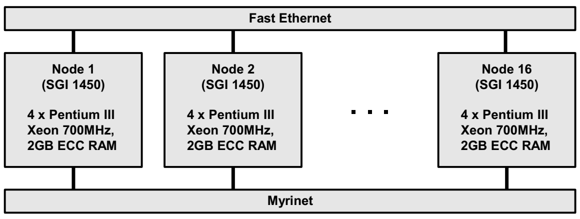


Figure 12. Experimentation platform. Gescher cluster has 16 SMP nodes. Each node has four processors.

3.2 Case study

In this section we demonstrate the usefulness of Performance Prophet by modeling and simulating a real-world material science program. For our case study we use LAPW0, which is a part of WIEN2k package [13]. LAPW0 calculates the effective potential within a unit cell of a crystal. The code of LAPW0 program is written in FORTRAN 90 and MPI [16]. LAPW0 comprises about 15,000 lines of code.

LAPW0 is executed in SPMD fashion (all processors execute the same program) on a multiprocessor computing system. A domain decomposition approach is used for the parallelization of LAPW0 (see Figure 11). The unit of material, for which LAPW0 calculates the effective potential, comprises a certain number of atoms (*NAT*). Atoms are evenly distributed to the available processes. This means that each process is responsible for calculation of the effective potential for a subset of atoms. For *NP* available processes, each process obtains *PNAT* = *NAT*/*NP* atoms.

Figure 12 depicts the architecture of Gescher cluster, which is located at Institute of Scientific Computing, University of Vienna [2]. Gescher comprises 16 SMP nodes. All nodes of Gescher are of type SGI 1450. Each node of the cluster has four Pentium III Xeon 700 MHz processors, and 2GB ECC RAM. The nodes of Gescher are interconnected via a 100Mbit/s Fast Ethernet network and a Myrinet network. For our experiments we have used the Fast Ethernet network. Gescher serves as our platform for performance measurement experiments of LAPW0.

In what follows in this section we develop and evaluate the model of LAPW0 with Performance Prophet. We validate the model of LAPW0 by comparing simulation results with measurement results for two problem sizes and four system configurations.

Figure 13 illustrates the procedure for the development of performance model of LAPW0 with Performance Prophet. Due to space limitations it is depicted just a fragment of the UML model of LAPW0. We developed the model of LAPW0 by using the modeling elements that are available in the toolbar of Performance Prophet. Basically, Performance Prophet permits to associate to each modeling element a cost function. A cost function models the execution time of the code block that is represented by the performance modeling element. Figure 13 depicts the association of cost function *CalcMPM* to action *Calculate Multipolmoments*. This cost function was generated based on measurement data by using regression. Regression is a technique for fitting a curve through a set of data values with some goodness-of-fit criterion.

We validated the performance model of LAPW0 by comparing simulation results with measurement results for two problem sizes and four system configurations. The problem size is determined by the parameter *NAT*, which indicates the number of atoms in a unit of the material. We have validated the performance model of LAPW0 for *NAT* = 32 and *NAT* = 64. The system configuration is determined by the number of nodes and the number of processing units. We have validated the performance model of LAPW0 for

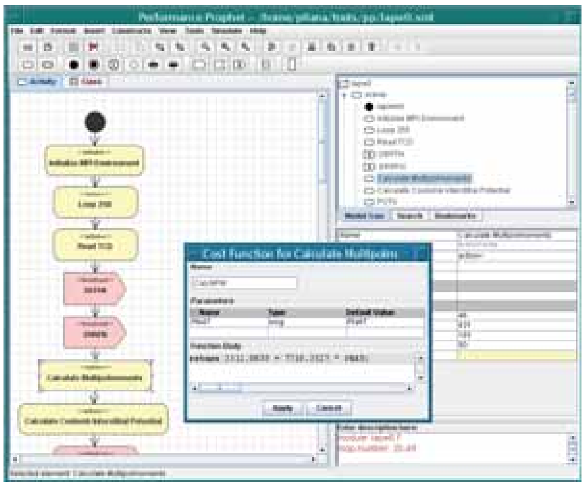

Performance Prophet
Cost Function for Calculate Multipolm
Calculate Multipolmoments
Apply
Cancel

Figure 13. Modeling of LAPW0 with Performance Prophet.

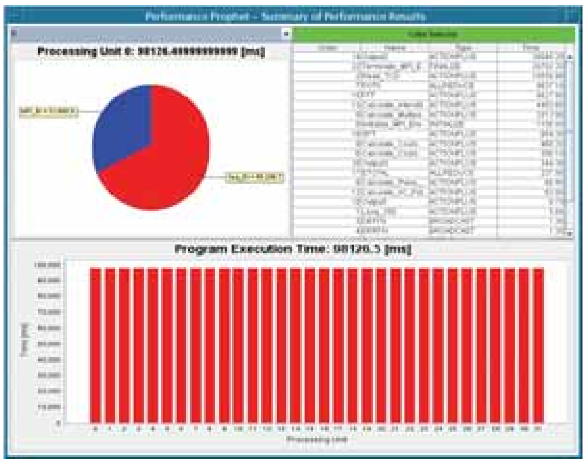

Performance Prophet – Summary of Performance Results
Processing Unit 0: 98126.4999999999 [ms]
Program Execution Time: 98126.5 [ms]
Processing Unit

Figure 14. Visualization of performance prediction results for LAPW0. Results are obtained from the simulation of execution of LAPW0 on eight four-processor nodes for problem size 32 atoms. One process is mapped to each processor (total 32 processes).

NAT = 32					
System	*Ts* [*s*]	*Tm* [*s*]	*Te* [*s*]	*Tm*/*Te*	*Error*[%]
N1P4	280	264	0.01	26,400	6
N2P8	170	166	0.02	8,300	2
N4P16	126	131	0.04	3,275	3
N8P32	98	113	0.08	1,413	13
NAT = 64					
System	*Ts* [*s*]	*Tm* [*s*]	*Te* [*s*]	*Tm*/*Te*	*Error*[%]
N1P4	543	501	0.01	50,100	8
N2P8	314	264	0.02	14,700	7
N4P16	211	197	0.04	4,925	7
N8P32	184	164	0.09	1,822	12

Table 1. Simulation and measurement results for LAPW0. In *NxPy*, *x* denotes the number of nodes *N*, and *y* denotes the total number of processes *P*. *Ts* is simulated time, *Tm* is measured time, and *Te* evaluation time. All times are expressed in seconds [*s*].

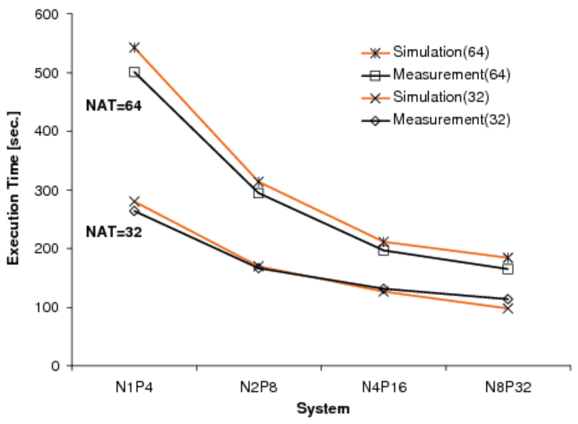


Figure 15. Simulation and measurement results for LAPW0.

the following system configurations: one node and four processes (N1P4), two nodes and eight processes (N2P8), four nodes and 16 processes (N4P16), eight nodes and 32 processes (N8P32). Each node comprises four processors. On each processor is mapped one process.

Table depicts simulation and measurement results for LAPW0. The second column of table, which is indicated with T_s, shows the performance prediction results for LAPW0 that we have obtained by simulation. Measurement results of LAPW0 are presented in the third column, which is indicated with T_m. The column that is indicated with T_e presents the CPU time needed for evaluation of the performance model of LAPW0 by simulation. All simulations were executed on a Sun Blade 150 (UltraSPARC-IIe 650MHz) workstation. We compare the time needed to execute the real LAPW0 program on our SMP cluster with the time needed to evaluate the performance model on a Sun Blade 150 workstation in the column that is indicated with T_m/T_e. We may observe that model-based performance evaluation of LAPW0 with Performance Prophet was several thousand times faster than the corresponding measurement-based evaluation. The rightmost column of the table shows the percentage error, which serves to quantify the prediction accuracy of Performance Prophet. We have calculated the percentage error using the following expression,

$$Error[\%] = \frac{|T_s - T_m|}{T_m}100,$$

where T_s is the simulated time and T_m is the measured time. We may observe that the prediction accuracy of Performance Prophet for LAPW0 was between 2% and 13% (see Table 1). The average prediction accuracy was 7%. Simulation and measurement results for LAPW0 are graphically presented in Figure 15.

4 Conclusions

The performance-oriented program development for large-scale computing systems is a time-consuming, error-prone, and expensive process that involves many cycles of code editing, compiling, executing, and performance analysis. This problem is aggravated when the program developer has access to only a part of the computing system resources and for only a limited time. The limited access to large-scale computing systems is a common practice, because the resources of this kind of systems are shared among many users. The model-based performance analysis may be used to overcome these obstacles.

In this paper we have presented a hybrid approach for the development of high-level performance models of large-scale computing systems, which combines mathematical modeling and discrete-event simulation. For the purpose of evaluation of our approach we have developed Performance Prophet, which is a performance modeling and prediction system. Performance Prophet provides a UML-based GUI, which alleviates the problem of specification and modification of the performance model. Based on the user-specified UML model of a program, Performance Prophet automatically generates the corresponding performance model and evaluates it by simulation. We have demonstrated the usefulness of Performance Prophet by modeling and simulating LAPW0, which is a real-world material science program that comprises about 15,000 lines of code. In our case study, the model evaluation with Performance Prophet on a single processor workstation was several thousand times faster than the execution time of the real program on our SMP cluster. We validated the model of LAPW0 by comparing the simulation results with measurement results for two problem sizes and four system configurations. The average prediction accuracy was 7%.

In future we plan to investigate the applicability of our approach for performance prediction of Grid workflows.

Acknowledgements

The work described in this paper was partially supported by the Austrian Science Fund (FWF) under the project AURORA.

References

[1] L. Dagum and R. Menon. OpenMP: An Industry-Standard API for Shared-Memory Programming. *IEEE Computational Science and Engineering*, 5(1):46--55, Jan./Mar. 1998.

[2] Gescher Cluster. University of Vienna, Institute of Scientific Computing. http://gescher.vcpc.univie.ac.at.

[3] S. Graham, M. Snir, and C. Patterson. *Getting Up to Speed: The Future of Supercomputing*. The National Academies Press, 2004.

[4] D. Kerbyson, A. Hoisie, and H. Wasserman. Use of Predictive Performance Modeling During Large-Scale System Installation. *Parallel Processing Letters*, 15(4), December 2005.

[5] Mesquite Software: CSIM. http://www.mesquite.com.

[6] Object Management Group (OMG). UML 2.0 Superstructure Specification. http://www.omg.org, August 2005.

[7] Parallel Virtual Machine (PVM). http://www.csm .ornl.gov/pvm/.

[8] R. Paul. Activity Cycle Diagrams and the Three Phase Approach. *Winter Simulation Conference (WSC 1993)*. Los Angeles, California, United States, 1993. IEEE.

[9] S. Pllana, I. Brandic and S. Benkner. Performance Modeling and Prediction of Parallel and Distributed Computing Systems: A Survey of the State of the Art. *The Fifth International Conference on Complex, Intelligent and Software Intensive Systems - 3PGIC Workshop*. Vienna, Austria, April 2007. Copyright (C) IEEE Computer Society.

[10] S. Pllana and T. Fahringer. PerformanceProphet: A Performance Modeling and Prediction Tool for Parallel and Distributed Programs. *The 2005 International Conference on Parallel Processing (ICPP-05). Performance Evaluation of Networks for Parallel, Cluster and Grid Computing Systems.* Oslo, Norway, June 2005. IEEE Computer Society.

[11] S. Pllana and T. Fahringer. UML Based Modeling of Performance Oriented Parallel and Distributed Applications. *Winter Simulation Conference (WSC 2002)*. San Diego, California, USA, December 2002. IEEE.

[12] Rice Simulator for ILP Multiprocessors (RSIM). http://rsim.cs.uiuc.edu/rsim/.

[13] K. Schwarz, P. Blaha, and G. Madsen. Electronic structure calculations of solids using the WIEN2k package for material sciences. *Computer Physics Communications*, 147:71--76, 2002.

[14] H. Schwetman. Hybrid Simulation Models of Computer Systems. *Communications of the ACM*, 21(9):718--723, 1978.

[15] H. Schwetman. CSIM19: A Powerful Tool for Building System Models. *Winter Simulation Conference (WSC 2001)*. Arlington, VA, USA, December 2001. ACM.

[16] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra. *MPI: The Complete Reference*. MIT Press, 1998.

[17] TOP500 Supercomputer Sites. http://www.top500.org.

Corresponding author: Sabri Pllana
University of Vienna
Institute of Scientific Computing
Nordbergstraße 15/C/3, 1090 Vienna, Austria
pllana@par.univie.ac.at

Received: June 23, 2007
Revised: September 12, 2007
Accepted: October 3, 2007

Optimal Multivariable Control Design Using Genetic Algorithms

M. Atanasijević-Kunc, A. Belič, R. Karba, Faculty for Electrical Engineering, Ljubljana, Slovenia

In the paper the binary semibatch-rectification process is presented. Nonlinear model is first used for the evaluation of different operating strategies. Later on, one of the possibilities is significantly improved using closed-loop operation, where several important goals where taken into account. For this purpose multivariable controllers were designed in several steps and compared regarding their complexity and quality of process operation. The best results, regarding the mass and the quality of the final product, were obtained using MIMO nonlinear, time-varying PID controller, which was optimized with genetic algorithms.

SNE 17/3-4, December 2007

Introduction

For the separation of mixed components on the basis of volatility, different kinds of distillation are used in chemical industry [1, 2, 7]. The process under consideration is shown in Figure 1. It consists of the reboiler, column and condenser. In the reboiler the mixture is heated and the vapour goes up through the column, giving up part of its energy at different levels. For good liquid - vapour contact the column is filled with the so called plates or trays which are during operation completely full of liquid at boiling temperature. Through this process the composition of more volatile component is increasing along the height of the device. In the condenser sufficient heat exchange must occur to ensure complete condensation. However, there is no need to cool the distillate, because part of the liquid is returned to the top of the column as the reflux flow at a temperature near to the boiling point. This is realized by the reflux distributor, which returns one part of the liquid from the condenser to the column, while the other part is collected as a final product.

The distillation devices are in most cases used as continuous systems where the mixture is fed continuously into the column at the prescribed position somewhere along the column. In the case of small production and frequently changing mixture, however, batch rectification (distillation is often called rectification for the batchtype regime of column operation) is suitable. Here, the mixture is put into the reboiler and the separation procedure lasts till the concentration and/or mass of the mixture in the reboiler becomes so small that the prescribed quality of the distillate can no longer be assured. The fact that most impurities in the mixture remain in the reboiler and thus the column is not soiled, as in continuous operation, can also be regarded as an advantage.

In some cases this type of distillation can be improved by the use of semibatch rectification. In such cases the reboiler is continuously fed with the same flow as is the flow of the distillate which represents the final product. The advantages of this type of distillation are the enlarged mass of the final product and the constant level of the mixture in the reboiler which ensures approximately constant vapour flow rate. It is also important to notice that each start-up of the system is very time - consuming procedure.

Regarding semibatch rectification, as is used in our case, two main operating possibilities are known: with constant and variable reflux. With the developed model both possibilities are evaluated regarding the mass of the product and its quality in the second section of the paper. In the third section further improvement of the system is suggested through the comparison of different control strategies, where the best results were achieved using multivariable, nonlinear, time-varying controller.

During modelling and control design the usage of different optimization procedures is very frequent. When the problems are complex and a great number of parameters are treated, genetic algorithms, as stochastic optimization technique, have great potential. Such situations can often be met with multivariable systems. Here the complexity is the result of system and controller dimensions while in the same time several design goals must be taken into account. Relative disadvantage of such optimization is the great number of input parameters which influence the final result. To overcome some of the mentioned problems design flow was organized in such manner that design results from previous steps were used for optimization problem definition in the next.

1 Problem description

As mentioned, our system operates as semibatch rectification process where its main task is to separate methanol from water with impurities. Repeated usage of dissolvents means substantive profit for the indus-

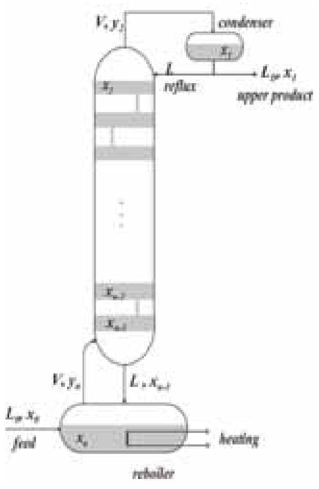


Figure 1. Semibach rectification system

try. The main goal of the operation of each batch can be oriented mainly to obtain very high concentration product or to produce as much of final product as possible where the concentration of the distillate can be lower. With our study we tried to evaluate the optimal regime of operation taking into account both aspects. For this the mathematical model was developed, where the following assumptions were taken into account: the mass of the mixture in the whole system is constant, as the flow of the feed $L_0(t)$ and the flow of the final product $L_D(t)$ are kept equal, the vapour flow rate $V(t)$ and the reflux $L(t)$ are equal through the whole height of the column and the composition of mixed components is homogeneous in the plates, condenser and reboiler. Mass balance equations for more volatile component can in this case be described as follows. For the condenser:

$$M_1 \frac{dx_1(t)}{dt} = V(t)\, y_2(t) - V(t)\, x_1(t), \tag{1}$$

for the i-th plate:

$$M_i \frac{dx_i(t)}{dt} = V(t)\, y_{i+1}(t) - V(t)\, y_i(t) + L(t)\, x_{i-1}(t) - L\, x_i(t) \tag{2}$$

and for the reboiler:

$$M_n \frac{dx_n(t)}{dt} = L_0(t)\, x_0(t) - V(t)\, y_n(t) + L(t)\, x_{n-1}(t) \tag{3}$$

where M_i are mass holdups, where index $i = 1$ is used for the condenser and $i = n$ for the reboiler, while $x_i(t)$ and $y_i(t)$ are the concentrations of more volatile component in liquid and vapour phase and $x_0(t)$ is the concentration of the feed. In our case the system has 6 plates and is therefore of the 8th order and highly nonlinear.

The relationship between liquid (x_i) and vapour (y_i) equilibrium at each plate can be estimated using corresponding tables [8] (see Fig. 2). But as tabled data are obtained for optimal conditions, in the model this values have to be decreased taking into account the level of the plate and its efficacy. The parameters were therefore defined using available data and finally adjusted to match the measurement data which were obtained by two main experiments as are illustrated in Figs. 3. During the first experiment (the system start-up is not included) both flow-rates $L(t)$ and $V(t)$ were constant, while during the second experiment the reflux was increased by the operator, taking into account the measured temperature in the condenser. The estimated hold-ups were in our case the following: M_1 = 105.99, M_2 = 5.35, M_3 = 5.33, M_4 = 5.29, M_5 = 5.22, M_6 = 5.08, M_7 = 4.84, M_8 = 5441.13, while the efficacy factors were defined us: α_2 = 1, α_3 = 0.7914, α_4 = 0.7804, α_5 = 0.7453, α_6 = 0.7191, α_7 = 0.6419, α_8 = 0.6419. The concentration of the feed was approximately constant and equal 0.315. With this model very good matching with measurement data was obtained.

In Fig. 3a both flow-rates are presented, while in Fig. 3b to Fig. 3i the concentrations in liquid (solid lines) and vapour (dotted lines) phase are illustrated. In Fig. 3j the concentration of the final product is indicated with the respect to two interesting values, 75% and 80%, while in Fig. 3k the mass of the final product for both experiments is presented. As in the first case the difference between both flow-rates is kept constant, the quantity of product is increasing linearly. But its concentration is decreasing and reaches the value of 75% after 4217 seconds. If higher quality is important, ti. 80%, the production should be stopped after 1526 seconds, so that corresponding masses would be 966 or only 349. In the second case, where the reflux is not constant, the production is slower but

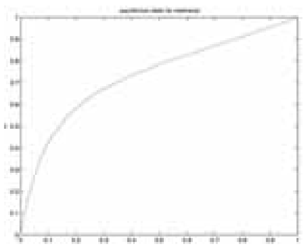


Figure 2. Equilibrium data for methanol

in this case the mass of the product with concentration 0.75 can reach the value 3991, while in the case of 80% the value is 355. This results indicate the need of suitable control action.

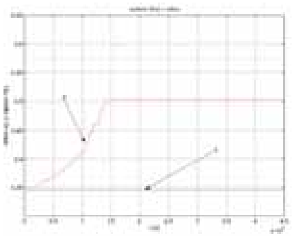
2 Control design

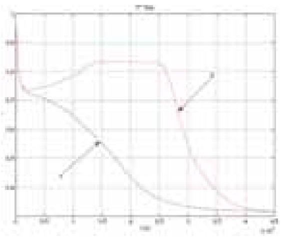
Design strategy was realized taking into account the following system properties:

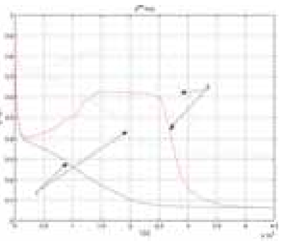
- in spite of the fact that the concentration of more volatile component in the condenser is very important, this variable was not chosen to be controlled; better choice proved to be $x_2(t)$, as the hold-up of the tray is significantly smaller and disturbance effects can be detected earlier; on the other hand constant concentration at the top of the column guarantees also constant quality of the final product;
- distillation columns are very sensitive regarding the activity of the reflux which can also flood the device; this means that the equilibrium between the liquid and vapour phase is completely destroyed and the batch must be stopped; this indicates the need of the second control input, $V(t)$;
- as the second output the concentration in the bottom of the column $x_7(t)$ was chosen; if this variable is kept on approximately constant value, also the disturbance of decreasing concentration of the mixture in the reboiler would be reduced; this disturbance is very important for system dynamical properties and is the reason, that the process is not working at its equilibrium, as is usually the case with continuous devices [1,2];
- regarding the complexity of chosen controller simple solutions are always of great interest, but there is also the exciting question: how good can the result be?

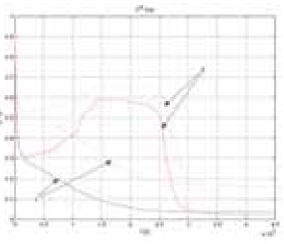
The design has stared with the analysis based on linearized models [4], obtained in different operating points. This study proved that system parameters and its dynamic properties are far from constant. So in the first step the proportional, robust MIMO-controller [10] was tuned in the following form:

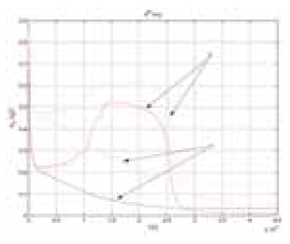
$$K_P = \begin{bmatrix} 0.5915 & -0.9860 \\ 0.3698 & -0.9856 \end{bmatrix} \qquad (4)$$

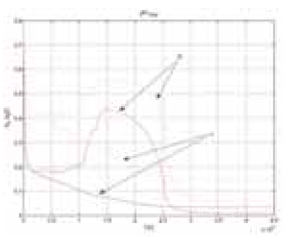

(a) $L(t), V(t)$

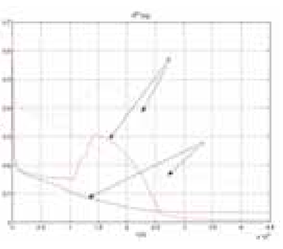

(b) $x_1(t)$

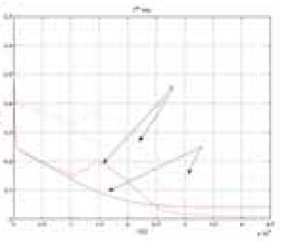

(c) $x_2(t), y_2(t)$

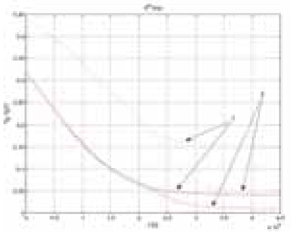

(d) $x_3(t), y_3(t)$

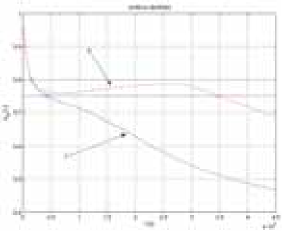

(e) $x_4(t), y_4(t)$


(f) $x_5(t), y_5(t)$


(g) $x_6(t), y_6(t)$


(h) $x_7(t), y_7(t)$


(i) $x_S(t), y_S(t)$


(j) $x_D(t)$

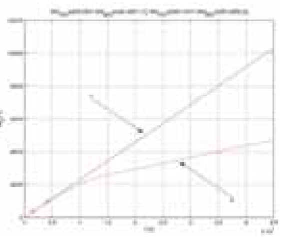

(k) $M_D(t)$

Figure 3.

The reasons for such a decision were the following:

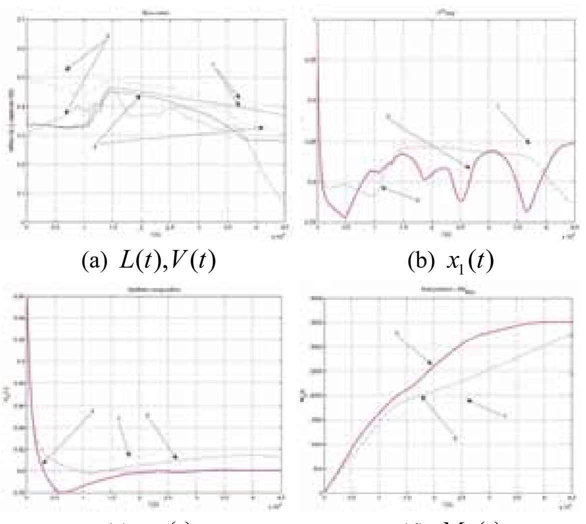
- First, it is the simplest solution regarding MIMO-control; the fact that such controllers do not reduce steady-state errors to zero is in our case not very serious problem, as with suitable chosen reference values the concentration of distillate can still reach the desired value.
- So during this first step attention was devoted also to the selection of adequate reference values;
- In addition, attention was devoted to the fact that both control signals have limited operating range and their difference should not be too large (less than 0.27) because of the so-called "flooding" problem.

The results are illustrated in Figs. 4 with dotted lines and with the number 1. Reference value of the first output was chosen as 0.85 and of the second as 0.48. From the presented responses the following can be concluded. With all limitations satisfied, the final product is during the whole batch process (12.5h) higher than 80%, while in the same time the mass of the product increased to the value 2433.6 which represents almost seven times more (6.85) than in the case where the reflux was controlled by the operator. This solution therefore represents significant improvement and can in addition decrease also the disturbances caused by outside temperature changes or feed concentration changes.

For additional noticeable improvement of the result system's changing properties have to be taken into account. As they are closely related with the duration of the batch process we have decided to test different time-varying nonlinear controllers in comparison with presented result. Here the efficacy of two types of controllers is given. The first has P and the second PID-structure. As the whole batch was divided into 10 segments, this means that each parameter of the controller can be described with the vector of 11 elements. For the P-structure this represents the problem with 44 and for the PID-structure the problem with 133 parameters as time constant of the D-part was chosen to be equal for all elements and constant. Due to these large numbers the solutions were optimized using genetic algorithms (GA) [5, 9]. The most important influence to searching procedure has also in this case a suitable chosen criterion or fitness function [3]. In addition we have tried to incorporate the knowledge of previous design in such manner that the result after optimization can not be worse than already known one. So the following fitness function was applied:

$$J = \begin{cases} 0 & \max(V - L) > 0.27 \\ 0 & \max(x_1) < 0.75 \\ 7\left(\int e_1^2(t)\,dt + \int e_2^2(t)\,dt + \int M_d(t)\,dt \right) & else \end{cases} \qquad (5)$$

The last condition in (5) was chosen also regarding the fact that all three integral expressions have approximately the same values for the first solution. In addition the starting generation (of 50 individuals or chromosomes) was not chosen randomly but was defined with individuals from the previous solution. The first generation had therefore all elements equal, while during evolution one best chromosome was automatically transferred to the next generation. After 50 generations we have obtained the results which are illustrated in Figs. 4 and 5. In Figs. 5 the controller parameters are illustrated, while system responses are presented in Figs. 4 with dashed lines for the P and with solid lines for the PID-structure. Observing the concentration and mass of the final product the following can be concluded: the second controller has increased the product for 33% and the third for 44% regarding the first one. In the case of PID-control the batch can finish approximately two hours earlier which therefore seems to be the best of presented solutions regarding the quality of control. The complexity of the last two solutions if of course grater but is not very serious problem in the case of computer control. When using classical industrial controller more simple and robust solutions are preferred.



(a) $L(t), V(t)$ (b) $x_1(t)$

(c) $x_d(t)$ (d) $M_d(t)$

Figure 4.

3 Summary

In the paper the nonlinear model of semibatch distillation column is presented which was developed for control design purposes. For this system three different control strategies are compared. The first is robust MIMO-P controller while the other two are time - varying MIMO nonlinear controllers with P and PID - structure and were tuned using GA optimizaton. The design flow was oriented in such manner that the information from previous design steps was taken into account in the next one which has shortened the optimization procedure. We have to mention that all design and simulation results were obtained in MATLAB [6].

References

[1] M. Atanasijević, R. Karba, B. Zupančič, F. Bremšak, *Modelling and Simulation of Dynamic Behaviour of Industrial Continuous Distillation Column*, Proceedings of 6th Mediterranean Electrotechnical Conference, Ljubljana, pp. 868-826, 1991.

[2] M. Atanasijević, R. Karba, B. Zupančič, F. Bremšak, *Modelling and Simulation Apgroach to Structural Rearrangement of Distillation Column*, Proceedings of the 13th IMACS World Congress on Computation and Applied Mathematics, Dublin, pp. 879-882, 1991.

[3] M. Atanasijević, R. Karba, B. Zupančič, A. Belič, *The use of genetic algorithms in modelling the multivariable control problem*. Proceedings of Eurosim '98 Simulation Congress, Vol. 3, pp. 548-552, 1998.

[4] M. Atanasijević, R. Karba, *Analysis Toolbox stressing parallelism of SISO and MIMO problems*, Preprints of the 15th World Congress, IFAC, Barcelona, Spain, 2002.

[5] M. Gen, R. Cheng, *Genetic Algorithms and Engineering Design*, John Wiley & Sons, Inc., 1997.

[6] *Matlab, The Language of Technical Computing*, Version 7, The MathWorks Inc., 2004.

[7] D. Matko, R. Karba, B. Zupančič, *Simulation and Modelling of Continuous Systems, A Case Study Approach*, Prentice Hall International Series in Systems and Control Engineering, 1992.

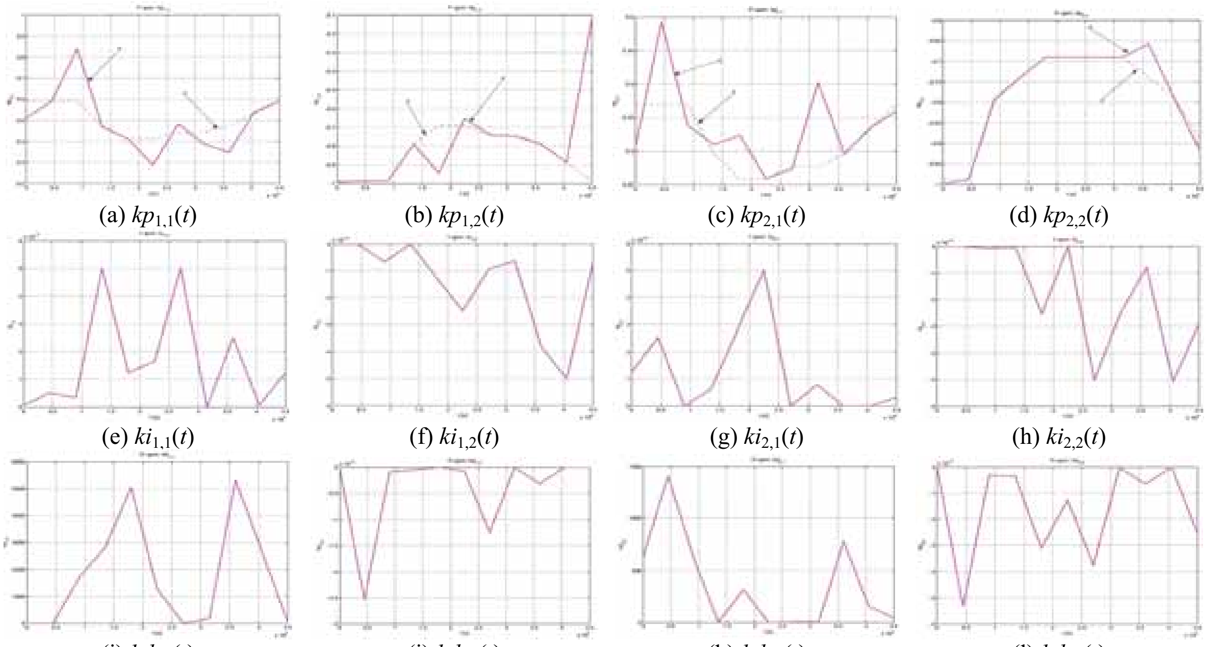
[8] R.H. Perry, C.H. Chilton, Chemical Engineer's Handbook, McGraw-Hill, NY, 1973.

[9] J.W.D. Prong van Hoogeveen, *Genetic Algorithms Toolbox*, TU Delft, 1995.

[10] S. Skogestad, I. Postlethwaite, *Multivariable Feedback Control, Analysis and Design*, John Wiley & Sons, Ltd, 2005.

Corresponding author: M. Atanasijević-Kunc,
Faculty for Electrical Engineering
University of Ljubljana, Slovenia

Accepted MATHMOD 2006, January 2006
Received: June 10, 2006
Revised: January 15, 2007
Accepted: January 20, 2007



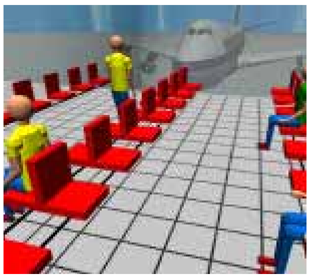
(a) $kp_{1,1}(t)$ (b) $kp_{1,2}(t)$ (c) $kp_{2,1}(t)$ (d) $kp_{2,2}(t)$

(e) $ki_{1,1}(t)$ (f) $ki_{1,2}(t)$ (g) $ki_{2,1}(t)$ (h) $ki_{2,2}(t)$

(i) $kd_{1,1}(t)$ (j) $kd_{1,2}(t)$ (k) $kd_{2,1}(t)$ (l) $kd_{2,2}(t)$

Figure 5.

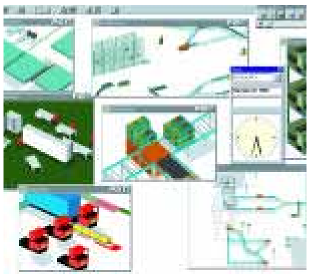
Don’t Speculate... Simulate!



You are operating in a complex and unpredictable business environment where balancing capacity defines your performance and profits. You deal with a lot of uncertainty and have to rework your planning continuously. Then Enterprise Dynamics simulation and control software is the ultimate powerful decision and planning tool to balance your resources. Enterprise Dynamics gives you an accurate image of your business processes and insight into utilization and yield of your resources, effectiveness, and logistical performance.

Enterprise Dynamics®, built by the people that brought Taylor to the market, is today's leading simulation software. Our software combines the powerful Enterprise Dynamics® Engine and many ready-to-use building blocks grouped into Enterprise Dynamic Suites for specific lines of business. Enterprise Dynamics® meets the latest standards in dynamic engineering and can be integrated into your existing system.

Incontrol Enterprise Dynamics
· The Netherlands +31 346.552500
· Germany, +49 203.3051.1170
info@EnterpriseDynamics.com



ENTERPRISE DYNAMICS Simulation Software®

Download your free evaluation version at:

www.EnterpriseDynamics.com

SNE 17/3-4, December 2007

SHORT NOTES

Particle Dynamics Simulation at Atomic Scale: Molecular Dynamics

Lukas D. Schuler, xirrus GmbH, Switzerland, *lukas.schuler@xirrus.ch*

In this introduction we recognize Molecular Dynamics (MD) simulation as an established method. To bridge the gap between the atomic world of substances and their measurable physical properties in the visible world, MD has been developed since 50 years [1, 2]. The main challenges of MD simulations are to calculate – through the means of statistical mechanics – thermodynamic properties of (or within) liquids [3] and give us insight and explanation of the dynamics. We provide some examples for explanation.

Introduction

In nowadays' practice, we are able to use Molecular Dynamics (MD) simulation to

- examine the dynamics of individual atoms, simple molecules, or larger structures in interaction,
- perform structural refinement of experimentally measured data,
- investigate changes in structure, e.g. protein folding.

Most often, we apply MD to molecules in solution. Some experimental data has been measured, but this did not explain all the details. MD provides then insight into the invisible, i.e. not measurable situations and processes.

1 Molecular model

A molecular model is based on the choice of

- atomic particles,
- interactions between particles,
- propagation dynamics,
- boundary conditions.

1.1 Atomic Particles

The elementary particles defined in classical Molecular Dynamics are atoms. An MD atom corresponds to a chemical element of the periodic table, consisting of its mass, a partial charge (if any), its current position and its velocity in 3D-space.

1.2 Interactions between particles

Non-bonded terms

Figure 1 shows that atoms interact strongly repulsive if they overlap. At intermediate separation, the interaction is attractive, and at larger separation, atoms lose contact and cannot notice each other. The parameter σ can be thought of as the size of an atom, and ε as the fluffiness of the electron cloud. These parameters are not determined a priori by some physical facts and have to be derived and optimized specifically for the simulation model.

For charged particles, an electrostatic potential is added to the potential energy function. They sum up to the non-bonded potential V as a function of the distance r_{ij} between particle i and j, the respective parameters for σ_{ij} and ε_{ij} and their individual partial charges q_i and q_j (eq. 1). ε_0 is the electric field constant.

$$V = \underbrace{4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right]}_{\text{Lennard-Jones Potential (Figure 1)}} + \underbrace{\frac{q_i q_j}{4\pi\varepsilon_0}\left[\frac{1}{r_{ij}}\right]}_{\text{Electrostatic Potential}} \tag{1}$$

Bonded terms

If atoms are connected by chemical bonds, the non-bonded interaction is replaced by bonded interactions. Dynamics of chemical bonds (vibrations), however, are usually not of interest. So bonded atoms are constrained at a fixed bond length.

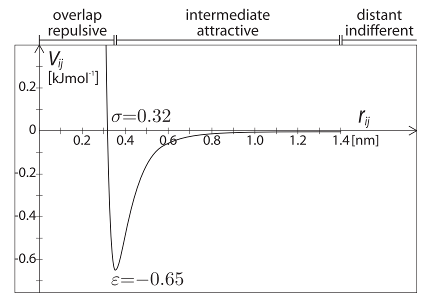


Figure 1. Example of a potential energy function for oxygen atoms of water molecules [4].

In analogy to the chemical bonds, additional interactions are introduced, that keep molecules in geometrical shape, e.g. for bond angles or torsions. These potentials are chosen to fulfill basic chemical characteristics, and convenient numerical properties instead of creating perfect physical representations of molecular situations.

The underlying parameter set determines how well experimental properties are reproduced.

1.3 Propagation dynamics

To advance the system for simulation, its particles are propagated in discrete time steps. For each time step in the range of femtoseconds (10^{-15} s) we apply the integration scheme:

- From the interaction functions, we derive the forces acting on individual particles according to their current positions.
- The forces provide us with the changes in velocities according to Newton's equation of motion.
- We derive new positions of the particles from their velocities.

This scheme is looped over the total simulation period.

1.4 Boundary conditions

During the simulation some global constraints are enforced. These are called boundary conditions.

Thermodynamic constraints. To obtain correct thermodynamic properties of the system, the propagation scheme must obey some restrictions: it shall be energy-conservative and time-reversible.

Spatial constraints. MD systems are confined in a small volume (of nanometer scale). Simulation in vacuum shows finite size effects – think of surface tension in a droplet, an effect we usually do not want to investigate.

To eliminate surface effects at all, we apply periodic boundary conditions. For example, a cubic box is virtually replicated in every dimension, so that the total system size is virtually infinite and boundaryless. For this to work properly, the box shall be large enough to separate any particle from its own effects in another replication.

Experimental restraints. Experimental methods such as *X-Ray crystallography* or *nuclear magnetic resonance (NMR)* are a means of obtaining structural data such as interatomic distances. Often, these data are not sufficient to derive full molecular structures. However, such data can be introduced into simulation as non-physical forces that drive simulated molecules to obey the experimental data.

Virtual restraints. We may also introduce any virtual potential that drives our molecule into a position or conformation we like to investigate. Virtual restraints can be subtracted after the simulation without affecting the results. This is a main advantage over experiment.

1.5 Time saving approximations

The expense to represent a system in simulation increases with the number of particles in it. Although increasing, computational power is the limiting factor for system size and simulated time. To make MD simulation more computationally efficient, we employ some tricks without negatively affecting the results, e.g.

Cut-off. The potential energy function is calculated below a certain distance only (i.e. 1.4 nm in Figure 1) without loosing much accuracy. This saves the need to calculate many interactions of negligible energy contribution.

Longer time steps advance the system further at the same computational expense. Still, the parts of interest moving fastest must be sampled smoothly.

United atoms are ball-shaped parts of molecules like hydrocarbons, e.g. CH_3 and CH_2 that are modeled as one atom instead of several [5]. This reduces the number of particles in the system.

2 Determination and validation of parameters

2.1 Parametrisation

Parameters like ε, σ and q in (1) need to be determined to yield simulation results in agreement with experiment. To do so, we used many linear, cyclic and branched hydrocarbon molecules to yield experimental densities. Think again of σ as the size of a particle. If we grow the particles confined in a box at constant volume, the pressure will rise. By adjusting the σ parameter of equation (1) for the corresponding united atoms [5], we are able to set the correct experimental density.

2.2 Validation

The obtained parameters were validated against the heat of vaporization, the energy necessary to evaporate the pure liquid (H_{vap}, eq. (2)), and the free energy of hydration, the free energy it takes to solve the substance in water (F_{hyd}, eq. (3)). Let me explain how we can calculate these macroscopic properties by microscopic simulation.

Heat of vaporization

$$H_{\text{vap}} = V_{(g)} - V_{(l)} + RT \qquad (2)$$

In equation 2, the average total potential molecular energy of the liquid phase $V_{(l)}$ is subtracted from the average total potential molecular energy in the gas phase $V_{(g)}$. $V_{(g)}$ basically lacks the intermolecular contacts, whose energy is extracted by the subtraction. RT, where R is the gas constant and T is the temperature, accounts for the volume expansion at evaporation according to the ideal gas equation.

We simulated a box of liquid molecules and calculated their potential energy. After that we expanded the coordinates of every individual molecule into a box of 100 times the size to avoid intermolecular contacts. We obtained the potential energy for the gas.

Free energy of hydration

$$F_{\text{hyd}} = \int_0^1 d\lambda \langle \partial E / \partial\lambda \rangle_\lambda \qquad (3)$$

Let us say a molecule that interacts with surrounding water is in state A ($\lambda = 1$). When its interactions with the water molecules are switched off, it is in state B ($\lambda = 0$). The thermodynamic coupling parameter λ is used to change state A into state B in small steps. Individual MD simulations are carried out at a number of different λ values and the average of the derivative of the total energy E with respect to λ is calculated. The free energy of hydration is then obtained by numerically integrating from $\lambda = 0$ to 1.

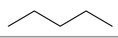
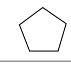
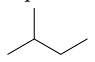
Units in [kJ/mol]	**Heat of vaporization**		**Free energy of hydration**	
	exp	sim	exp	sim
Pentane 	26.4	26.2	9.8	10.2
Cyclopentane 	28.5	27.7	5.0	5.1
Isopentane 	24.9	25.0	10.0	11.4

Table 1. Example results for hydrocarbon chains of 5 connected united-atoms. Taken from [5].

Table 1 shows that obtained results are in excellent agreement with experiment (within the accuracy reachable by MD simulation) and promising for the use of this parameter set for mixed hydrocarbons, and in water solution.

Experimentalists have collected data of similar branched molecules. The two molecules in Figure 2 differ by one united atom (CH_3). In simulation, the molecule to the left crystallizes, whereas the one to the right stays liquid at room temperature. This finding is again in agreement with experiment.

So far, we looked at *uncharged* atoms.

2.3 Polar molecules

Most biological or chemical molecules are built of additional atoms like oxygen, nitrogen, sulfur or phosphor. They carry partial charges what makes these molecules polar. E.g. oxygen atoms (O) are modeled with partial negative charge, such as in acetic acid (Figure 3).

Note that hydrogens that carry partial charges are treated explicitly and excluded from united atoms.

3 Lipid aggregate simulation

We performed the parametrisation in [5] with regard to lipid simulations. Lipids consist of a polar head group and a non-polar tail. The head group contains partial charges similar to Figure 3 and is attractive to water molecules, that do have partial charges as well. The tail consists of long hydrocarbon chains, similar to pentane in Table 1, and is uncharged. Fatty acids and soap are some examples of lipids.

When dissolved in water, lipids tend to aggregate, as the non-polar tails do not like being exposed to water. In experiment, lipid aggregation depends on the concentration of lipids and salt in aqueous solution, and on their size. In solution, nearly spherical aggregates of lipids are known as *micelles*. We simulated several lipid aggregates [7] under different conditions.

One micelle consisted of 90 lipids with a tail length of 12 carbons. Another one consisted of 64 of the

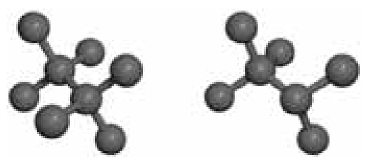


Figure 2. The molecules to the left are solid, the ones to the right liquid at room temperature.

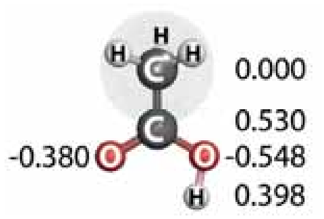


Figure 3. Acetic acid. The partial charges sum up to the total charge of the molecule, which is neutral (zero) here. The charge numbers shown are taken from glutamic acid in [6]. CH_3 is indicated as united atom.

same lipids. The aggregate containing more lipids was more stable in simulation. This is not the case if the lipid tail is shorter, containing only 8 carbon atoms. With 90 molecules in the micelle, its structure is not maintained. Some lipids start to diffuse completely into solution.

This finding was supported by experiment, which assumes an ideal aggregation for the shorter tails containing 43+/-5 lipids, way below 90.

Structural dynamics

We have analyzed whether the tail ends prefer to reside near the center or not, for the stable micelles. The lipid tail end (united atom type CH_3) appears up to close to the head group (oxygen atoms). This finding illustrates that lipids within these structures are able to expose their tails to the surface. This is not the case in simulations of the same lipids in lipid membranes – a more crystalline lipid aggregate structure.

The diffusion coefficient for the lipids was calculated from our simulation to be $2 \cdot 10^{-6} cm^2 s^{-1}$. It is in the expected range for other lipids measured experimentally. For comparison, this is about ten times slower than self-diffusion of water.

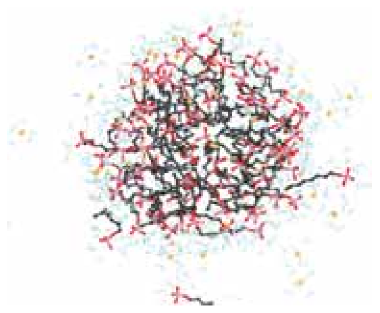


Figure 4: An aggregate of 90 lipid molecules and sodium ions with surrounding water molecules after 3 ns. Initially, all 90 lipids have been placed within the aggregate structure. Three lipids (to the lower left, right and at the bottom) have visibly left the micelle.

4 Conclusion

We use Molecular Dynamics simulation (MD) to investigate a vast variety of molecules on atomic resolution. With the appropriate choice of interaction parameters, we are able to

- reproduce experimental findings while providing atomic detail,
- explain phenomena not accessible by experiments,
- better understand the chemical and biological mechanisms that build our world.

Acknowledgements

We would like to acknowledge Philippe Hünenberger for his inspiring talk and Wilfred van Gunsteren inviting us to his 60th birthday symposium.

References

[1] B. J. Alder, T. E. Wainwright: *Phase transitions for hard sphere system*. Journal of Chemical Physics, 27, pp 1208-1209, 1957.

[2] B. J. Alder, T. E. Wainwright: *Phase transitions for hard sphere system*. In (I. Prigogine, ed.) Proc. International Symposium on Statistical Mechanical Theory of Transport Processes (Brussels, 1956), Interscience, New York, 1958.

[3] M. P. Allen, D. Tildesley: *Computer simulation of Liquids*. Clarendon Press, Oxford, 1987.

[4] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, J. Hermans: *Interaction models for water in relation to protein hydration*. In (B. Pullmann, Reidel eds.) Dordrecht, 1981.

[5] L. D. Schuler, X. Daura, W. F. van Gunsteren: *An Improved GROMOS96 Force Field for Aliphatic Hydrocarbons in the Condensed Phase*. Journal of Computational Chemistry, 22, pp 1205-1218, 2001.

[6] W.F. van Gunsteren, S.R. Billeter, A.A. Eising, P.H. Hünenberger, P. Krüger, A.E. Mark, W.R.P. Scott, I.G. Tironi: *Biomolecular Simulation: The GROMOS96 Manual and User Guide*. Vdf Hochschulverlag AG, Zürich, 1996

[7] L.D. Schuler, P. Walde, P.L. Luisi, W.F. van Gunsteren: *Molecular dynamics simulation of n-dodecyl phosphate aggregate structures*. European Biophysics Journal, 30, 330-343, 2001.

Corresponding author: Lukas D. Schuller
xirrus Simulation GmbH
Buchzelgstrasse 36, CH-8053 Zürich, Switzerland
lukas.schuler@xirrus.ch

Received: September 5, 2007
Revised: October 8, 2007
Accepted: October 15, 2007

Modelling Reference Price Effects – A Discrete Dynamic Programming Approach

Lisa Gimpl-Heersink, Jürgen Wöckl, Alfred Taudes, Vienna University of Economics, Austria
{Lisa.Gimpl-Heersink, Juergen.Woeckl, Alfred.Taudes}@wu-wien.ac.at

We study a dynamic optimal price setting technique which allows for demand side dynamics - demand is sensitive to the firm's pricing history. People buy products just because they are on sale and they are less likely to buy products after price went up. In this paper we provide a decision-support system for dynamic retail pricing and promotion planning. Consumers' purchase decisions are founded not only on the reaction to the current price, but also on deviations from a reference price formed on the basis of past purchases.

SNE 17/3-4, December 2007

Introduction

In traditional economics and operational models consumers are viewed as rational agents who make decisions based on current prices, income and market conditions. In a market with repeated interactions, such as frequently purchased consumer goods, customers' purchase decisions are also determined by past observed prices. In each period, a manager makes decisions that influence demand. Examples of such sales levers could be prices, sales-force incentives and advertisements. For means of simplicity we consider the simple model where the only sales lever is price. We are not considering customers' income and market conditions. Figure 1 illustrates, that current demand is determined by the currentprice, but also affected by past prices.

We now try to find a formulation that captures the effect of past prices on demand. We see that the carrier of price is not based on its absolute level but rather on its deviation fromsome reference level (resulting from pricing history). If the price of a brand is below itsreference price, the observed price is lower than anticipated, resulting in a perceived gain.This would make the brand more attractive and raise demand (people buy products just because they are on sale). Similarly, the opposite situation would result in a perceived loss, reducing the probability that the brand is purchased (people are less likely to buy products after price went up). An important consequence of internal reference price formation is that although frequent price discounts may be beneficial in the short run, they may damage the brand in the long run when households get used to these discounts and reference prices drop. The reduced price becomes anticipated and loses its effectiveness, whereas the nonpromoted price becomes unanticipated and would be perceived as a loss. From now on we call *reference price* the standard price against which consumers evaluate the actual prices of products they are considering. In other words it is the perceived price, or what you expect to pay when entering a store. Knowing about these reference price effects we now want to provide a decision-support system for dynamic retail pricing and promotion planning. We decide which price to set at the beginning of each period of a given finite planning horizon with the objective to maximize expected discounted profits of the entire planning horizon.

1 Model

In literature we observe several ways a reference price can be formed. One introduced by [1] is to operationalize reference price rp as the one-period lagged price p for a brand: $rp_t = p_{t-1}$. Another way could be summing past prices (e.g. [2]). Exponentialsmoothing (introduced in the adaptive expectations framework by [3]) is the most commonly used and empirically validated reference price mechanism in literature (e.g. [2, 4, 5, 6, 7]):

$$rp_t = \alpha(rp_{t-1}) + (1-\alpha)(p_{t-1}), \quad 0 \le \alpha < 1 \qquad (1)$$

with rp_t and p_t being reference price and observed price respectively for a brand in period t. α is called the memory parameter and captures how strongly the reference price depends on past prices. Lower values of α represent a shorter term memory; in particular if α = 0, the reference price equals past price as in [1]. We can also see α as a proxy for loyality. Estimated parameters range from $\alpha \in [0, 0.925]$. A new and more realistic approach developed by [8] is based on the assumption that there exists an asymmetric adaption of consumers' reference price. One would

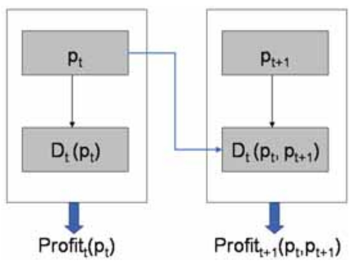


Figure 1. The effect of past prices on demand and profit

Figure 2. Symmetric and asymmetric demand model

expect that consumers adapt their reference prices with higher speed to price decreases, than to price increases (see figure 2):

$$rp_t = rp_{t-1} + \mu \max\{0, p_{t-1} - rp_{t-1}\} + \\ + \eta \min\{0, p_{t-1} - rp_{t-1}\}, \quad \mu \le \eta \tag{2}$$

It is easy to see that equation (1) is a special case of equation (2) with $\mu = \eta = (1-\alpha)$. We consider a monopolistic firm or create a monopolistic framework by restricting the price interval by an upper and a lower bound $p \in [\underline{p}, \bar{p}]$ which guarantees monopolistic behaviour again. Stochastic endogenous demands in consecutive periods are independent and their distributions depend on the item's price and the consumers' reference price in accordance with general stochastic demand functions.

$$D_t(p_t, rp_t, \varepsilon_t) = \beta_0 + \beta_1 p_t + \beta_2 \max\{p_t - rp_t, 0\} + \\ + \beta_3 \min\{p_t - rp_t, 0\} + \varepsilon_t \tag{3}$$

where ε_t stands for the random perturbation from the mean demand and $\beta_0 \ge 0$ and $\beta_1, \beta_2, \beta_3 \le 0$ are estimated parameters so that demand is a decreasing function. Without loss of generality we can assume that ε_t follows some distribution with mean zero and variance σ^2. Pricing decisions are made at the beginning of each period with the objective to maximize total expected discounted profit over the entire planning horizon:

$$\max_{p_t \in [\underline{p}, \bar{p}]} E\left[\sum_{t=1}^{T} \gamma^t (p_t - c_t) D_t(p_t, rp_t, \varepsilon_t)\right] \tag{4}$$

with $c_t \le p_t$ denoting the per unit ordering costs in period t, $\gamma \in [0,1]$ the discount factor and T the total number of periods of the finite planning horizon. Given this we can easily write the Bellman equation of the underlying dynamic programme.

$$v_t(rp_t) = \max_{p_t \in [\underline{p}, \bar{p}]} \{E[(p_t - c_t) D_t(p_t, rp_t, \varepsilon_t)] + \\ + \gamma E[v_{t+1}(rp_{t+1}(rp_t, p))]\} \tag{5}$$

$$v_{T+1}(rp_t) = 0$$

Note that $rp_{t+1}(rp_t, p)$ is known by equation (1) or (2). Given the recursive nature of the Bellman equation, we may compute the optimal values p_t and policy functions v_t using backward recursion.

2 Results

If equation (3) is symmetric with respect to the effect of gains and losses ($\beta_2 = \beta_3$), buyers are loss-neutral and the demand function is smooth. For loss-averse consumers the value function is steeper for losses than for gains ($\beta_2 < \beta_3$). In other words, a loss decreases value more than an equivalent sized gain would increase value. This is how we expect a rational consumer to behave. Nevertheless some people might still behave loss-seeking and evaluate a gain higher than a loss ($\beta_2 > \beta_3$). Figure 3 illustrates how different values of β_2 and β_3 affect the optimal price with respect to reference price. For $\beta_2 < \beta_3$ demand is declining more in the region $p_t - rp_t > 0$ than it is

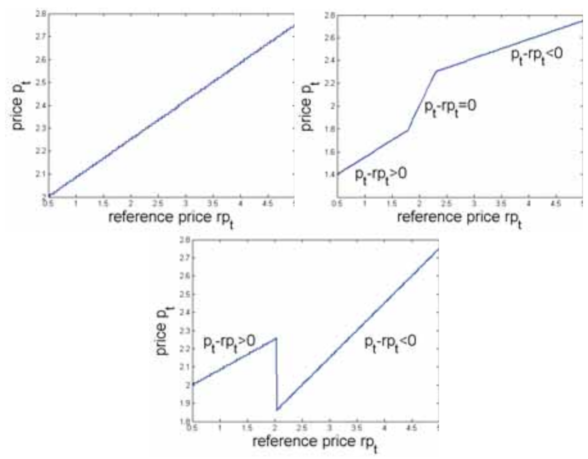


Figure 3. The impact of reference effect: Loss-neutral case ($\beta_2 = \beta_3$), loss-averse case ($\beta_2 < \beta_3$), loss-seeking case ($\beta_2 > \beta_3$).

Figure 4. Price path (loss-averse case)

Figure 5. Price path (loss-seeking case)

in the region $p_t - rp_t < 0$ which intuitively yields a greater positive slope for $p_t(rp_t)$. The same argument holds vice versa for $\beta_2 > \beta_3$.

According the approach provided by [2, 4, 5, 6, 7] we first take a closer look on loss-neutral and loss-averse consumers. Here the optimal prices converge to a steady state price. In the transient stage at the beginning of the time horizon, consumers either perceive discounts or surcharges until this steady state is reached. In literature we find useful analytical results for an explicit solution of the steady state of an infinite horizon model. They even provide monotonicity results for the transient stage. [7] for example shows transient monotonicity of reference price and

even finds transient monotonicity for price under certain conditions (see figure 4). If consumers' price is initially high, they will be led to perceive a gain in each period, as the firm will consistently price below the reference price. This perception of monotonic prices has the effect of a skimming strategy (if initial price expectations are high, start with a high price and keep decreasing it). Similarly, a low reference price (see figure 4) leads to a penetration-type strategy (if initial price expectation is low, start with a low price, then increase it).

These results are analytically extremely valuable but are not capable of giving a decision support system for dynamic retail pricing, as there we usually face finite time horizon models and thus can't be sure if a steady state is reached at all and if it is reached, when it is reached respectively. Besides, in practice it doesn't satisfy retail to know about price monotonicity but they rather need to know exactly which price to charge in each period in order to maximize total expected discounted profits. This shows the need for additional numerical simulation to existing analytical results. In finite horizon models we find theadditional discrepancy that we face a transient stage at the beginning and at the end of the planning horizon. The pricing policy at the end of the time horizon is always a markdown policy. Prices are drastically decreased (see figure 4) in order to raise demand and thus make as much profit as possible in the remaining time to go. Increasing the price expectation of consumers here is not reasonable, as the end of a product's life cycle is reached and the remaining future selling periods do not suffice to outweigh the loss of profit a price promotion would induce in the promotion period.

For loss-seeking consumers cyclical policies which follow a hi-lo pattern turn out to be optimal (see figure 5). In this case there exists no analytical solution for the price pattern which motivates numerical simulation again. The cycle-period equals $2ct$, for some positive integer c. We notice that the hi-lo range increases the greater the difference of consumer sensitivity to discounts and surcharges is. For high enough bounds $[\underline{p}, \overline{p}]$ on p the solution becomes degenerate, which means that the hi-lo values are always realized at the bounds $[\underline{p}, \overline{p}]$.

We further observe the phenomenon, which has never been discussed in known literature, that in the loss averse case it might under certain circumstances be better to charge the highest possible price in the first period, which raises the reference price for all future periods and thus outweighs the loss created in the first period. One of these circumstances could for instance be not to restrict the bounds on price to a level where one still observes concave demand for all possible prices. A degenerate solution, which loses the monotonicity property of price in reference price (see figure 7) and charges $\overline{p}$ at least in the first time period (see figure 6), is obtained. We observe that the region of reference prices for which the highest possible price is charged decreases over time and finally

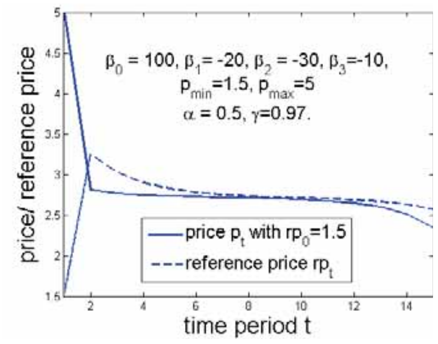


Figure 6. Degenerate price path

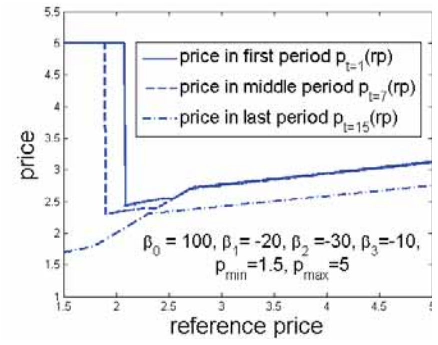


Figure 7. Nonmonotonicity

disappears in the last time period. This of course is intuitive as at the end of a product's life cycle there is no need for raising future reference prices.

Considering the more realistic asymmetric reference price model from equation (2) we observe that a non-cycling price policy is only optimal for $\beta_2 < \beta_3$, if the difference between η and μ is 'large enough'. 'Large enough' can't be formulated as a constant value, but depends on the hi-lo range at $\mu = \eta$. As the hi-lo range increases, $\eta - \mu$ has to increase to obtain a steady state pricing policy.

The great advantage of this numerical simulation is that it can easily be adapted to any other demand or reference price model. It is possible to drop unrealistic technical assumptions which are needed to solve the model analytically.

References

[1] L. Krishnamurthi, T. Mazumdar, S.P. Raj. *Asymmetric response to price in consumer brand choice and purchase quantity decisions*. Journal of Consumer Research, 19:387–400, 1992.

[2] R.S. Winer. *A reference price model of brand choice for frequently purchased products*. Journal of Consumer Research, 13:250–256, 1986.

[3] M. Nerlove. *Adaptive expectations and cobweb phenomena*. The Quarterly Journal of Economics, Vol. 72, No. 2. (May, 1958), pp. 227-240., 72(2):227–240, 1985.

[4] E. Greenleaf. *The impact of reference price effects on the profitability of price promotions*. Marketing Science, 14(1):82–104, 1995.

[5] P.K. Kopalle, A.G. Rao, J.L. Assuncao. *Asymmetric reference price effects and dynamic pricing policies*. Marketing Science, 15(1):60–85, 1996.

[6] G. Fibich, A. Gavious, O. Lowengart. *Explicit solutions of optimization models and differential games with nonsmooth (asymmetric) reference-price effects*. Operations Research, 51:721–734, 2003.

[7] I. Popescu, Yaozhong Wu. *Dynamic pricing strategies with reference effects*. Working paper, INSEAD, 2006.

[8] M. Natter, T. Reutterer, A. Mild, A. Taudes. *An assortment-wide decision-support system for dynamic pricing and promotion planning in DIY retailing*. Working paper, Vienna University of Business Administration and Economics, 2005.

Corresponding author: Lisa Gimpl-Heersink,
Institute for Production Management, Vienna
Univ.of Economics and Business Administration
1090 Wien, Austria
Lisa.Gimpl-Heersink@wu-wien.ac.at

Accepted ASIM 2006, June 2006
Received: October 10, 2006
Revised: May 10, 2007
Accepted: May 22, 2007

Parallel Transistor Level Simulation based on Parallel Equation Formulation implemented on a Beowulf Cluster

Bojan Anđelković and Vančo Litovski, Faculty of Electronic Automation, Serbia
abojan@elfak.ni.ac.yu, *vanco@elfak.ni.ac.yu*

First results obtained by a parallel transistor level simulation on a Beowulf type computer cluster are presented. The version of the simulator just developed implements parallelization in the equation formulation phase of the non-linear dynamic electronic circuit, only.

SNE 17/3-4, December 2007

Introduction

The simulation process of modern complex integrated electronic circuits may be characterized as memory and computationally intensive, and algorithmically complex. These properties pertain to the fact that a large number of ordinary (and, potentially, partial) nonlinear differential equations have to be solved for long running excitations. In addition, a huge amount of data may be created in a single simulation run, which needs to be processed and interpreted. Because of all these reasons simulation runtimes are very long. Having in mind that every design needs many simulation runs of the same design in order to get optimal solutions and satisfy the design requirements, it is obvious that long simulation runtimes lead to a slow design process. One possibility to reduce these runtimes is to parallelize the algorithm and use parallel computers to execute simulations. In this approach complex calculations necessary during the simulation process can be distributed over different workstations/processors and performed simultaneously.

Over last decade, as personal computers performance has increased and prices have fallen steeply, both for the PCs themselves and the network hardware necessary to connect them, dedicated clusters of PC workstations have provided significant computing power at low cost. One particular implementation of this approach, involving open source system software and dedicated networks, has acquired the name “Beowulf” [1].

The biggest obstacle to the widespread use of parallelism is inadequate software. Compilers that automatically parallelize sequential algorithms remain limited in their applicability. Best performance is still obtained when programmer himself implements the parallel algorithm that is the most appropriate for the specific domain problem.

There are several parallel computational models that define the types of operations available to the program. One of them that is the most common for applications on Beowulf clusters is message passing model of parallel computation, and in particular the Message Passing Interface (MPI) implementation of that model [2, 3]. In the message-passing model the complete program is split into a set of processes that have only local memory but are able to communicate with each other by sending and receiving messages.

There are several parallel simulators of electronic circuits that have been developed recently, such as Xyce [4], Titan [5] and SEAMS [6]. Titan and Xyce are parallel transistor level simulators that use SPICE as modeling language. Both simulators implement complex partitioning algorithms to split the circuit description and distribute generated partitions to different workstations/processors. These partitions are then simulated in parallel. Appropriate synchronization protocols should be applied to exchange necessary simulation data between the circuit partitions. The main goal of these parallel algorithms is to minimize communication between the workstations/processors and achieve their equal load. SEAMS is a VHDL-AMS simulator that implements parallel digital simulation, while parallel mixed-signal simulation is under the development. A broad survey of various parallel simulator implementations and algorithms can be found in [7].

This note presents the concept of development a parallel transistor level simulator of electronic circuits that executes on a Beowulf type cluster using MPI. Parallelization of equation formulation for nonlinear analog circuit elements is implemented in order to reduce simulation time.

1 Parallel Equation Formulation

In order to simulate complex mixed-signal electronic circuits at transistor level, they have to be modeled using algebraic equations and nonlinear Ordinary Differential Equations (ODE). ODEs are discretized in order to create sets of nonlinear algebraic equations. This generates a potentially large system of equations to be solved at a large number of time instants depending on the properties of the system under simulation and the stimulus signals. The system of nonlinear equations is solved iteratively with the help of linearization i.e. by application of Newton methods.

The algorithm for simulation of nonlinear dynamic electronic circuits in time domain is shown in listing 1 [8]. As it can be seen, at each iteration and at every time instant the matrix entries of the system of linear equations have to be recalculated. These entries are derivatives of the nonlinear equations and are computed within separate subroutines. Having in mind the number of matrix entries, the number of iterations and the number of time instants, it is necessary to provide an immense computational effort. It has been shown that even for small systems, equation formulation takes more computational time than equation solution. Therefore, the calculation of matrix entries and equation formulation for non-linear circuit elements can be parallelized. The part of the simulation algorithm that can perform in parallel is highlighted by a rectangle in listing 1.

If we consider the circuit matrix as a sum of several matrices the number of which is equal to the number of processors implemented, we may create the whole

```
generate node voltages and specified
    branch currents $x^0 = \left[(v^0)^T (i^0)^T\right]^T$;
choose $h$;
$n = 0$;

while($t < T$) {                    /* time loop */
    $m = 0$;
    predict $x^{n+1,0}$;

    until convergence {             /* iterative loop */
        generate descretized models;
        generate linearized models;
        formulate system of linear equations;
        solve the system and find $x^{n+1,m+1}$;
        $m$++;
    }
    $t = t + h$;
    $n$++;
}
```

Listing 1. The simulation algorithm for nonlinear dynamic circuits in time domain

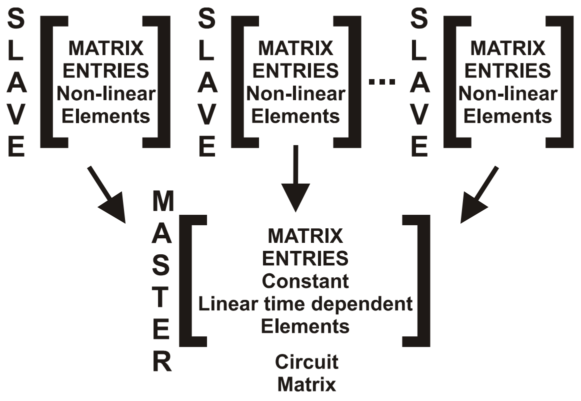

SLAVE
MATRIX ENTRIES Non-linear Elements
SLAVE
MATRIX ENTRIES Non-linear Elements
...
SLAVE
MATRIX ENTRIES Non-linear Elements
MASTER
MATRIX ENTRIES Constant Linear time dependent Elements
Circuit Matrix

Figure 1. Parallelization of equation formulation

matrix by creating its parts and then by summing them. That is illustrated in Fig. 1. There is no specific criterion for allocation of the circuit elements to specific processor (or submatrix). Simply the total number of non-linear circuit elements is divided by the number of processors and the list of elements is divided to equal parts and partitions are created. Such parallelization of the simulation algorithm is a new approach different from already developed solutions. It requires neither a sophisticated circuit and task partitioning algorithm nor synchronization protocols between these partitions, so it is easy to implement on a Beowulf cluster using MPI routines.

The generation of matrix entries for constant and linear dynamic elements is not performed in parallel, since these calculations may be performed outside of the iterative loop. Moreover, the matrix contributions for constant elements are calculated only once outside time and iterative loops, while entries for linear time dependent elements are calculated at every time instant outside iterative loop. All this reduces the overall time necessary for equation formulation. When parallel generation of matrix entries for various nonlinear elements is finished, the complete circuit matrix is formed (Fig. 1). Then system of linear equations is solved. That task can also be parallelized which should lead to further reduction in simulation time.

2 Parallel simulator implementation

The presented parallelization of equation formulation process is implemented in the simulator Alecsis [9]. It is a mixed-signal and mixed-domain simulator with proprietary hardware description language AleC++ [10] capable for modeling and simulation of complex systems containing different kinds of devices and subsystems [11]. The developed simulator with parallel simulation capability is called pAlecsis (*Parallel* Analog and Logic Electronic Circuits Simulation System).

The implementation of parallelization in the pAlecsis simulator on a Beowulf cluster is shown in Fig. 3. Parallel equation formulation is implemented using one of the most common of parallel algorithm prototypes, master-slave algorithm [2]. In this algorithm calculation of matrix contributions for non-linear circuit elements at each time instant and iteration is distributed to multiple slave processes and they are calculated simultaneously. At the same time master process calculates matrix entries for specific number of non-linear elements as well as for constant and linear time dependent elements. Master and slave processes execute on different cluster nodes (PC workstations). Since multiple cluster nodes calculate contributions for different elements in parallel, the time necessary for equation formulation decreases.

In order to minimize communication between cluster nodes, appropriate data structures for all elements of the circuit are generated on all nodes simultaneously during compilation of the AleC++ model. In that way all cluster nodes have the information necessary to generate matrix contributions for all elements. Each node of the cluster performs equation formulation and calculation of matrix entries for equal number of non-linear circuit elements. When entries for all elements on one slave are generated, they are sent to the master node using appropriate MPI routines (Fig. 2).

When the master node receives matrix entries from all slaves, it flushes them to the system matrix and performs one simulation step. In order to enable calculation of matrix entries on slave nodes, the master node should send to the slaves vectors of solutions of the system of equations for the two past time instants and previous iteration (denoted with *vp1*, *vp2* and *vi* respectively in Fig. 2). Appropriate MPI routines for transfering data are used to send and receive these vectors.

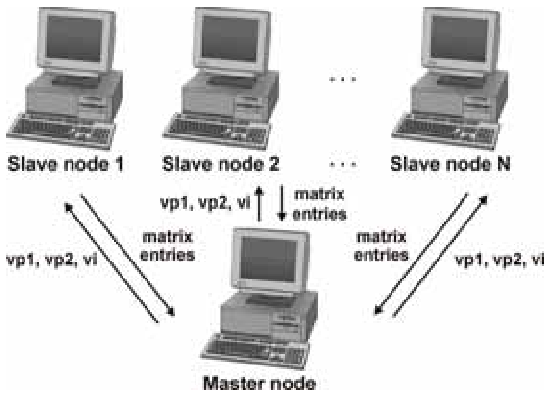


Figure 2. Implementation of the pAlecsis simulator on a Beowulf cluster

It is worth to mention here that the very solution of the system of simultaneous linear equations, that is generated in the way described above, *is not parallelized* yet. We intend to use SuperLU [12] for this purpose in the first next step of upgrading of pAlecsis.

3 Parallel Simulation Performances

Sequential simulation algorithms executing on a single workstation are tested for correctness usually by only seeing whether they give the right result. For parallel programs, that is not enough, but one wishes to reduce the simulation time. Therefore, measuring of simulation time is part of testing the parallel simulator to see whether it performs as intended. Usually performances of the parallel simulator are specifed as speedup. If parallel simulation executes on N single processor cluster nodes, speedup is normally defined as [2]:

$$Speedup = \frac{\text{Simulation time on 1 node}}{\text{Simulation time on N nodes}} \qquad (1)$$

Implemented parallel simulation algorithm reduces simulation time for bigger circuits when time necessary to calculate matrix entries for all elements at every time instant and every iteration exceeds time necessary to calculate matrix entries on slave nodes and send them to master node over the interconnecting network. For such circuits the parallel simulation on the cluster is faster than the simulation on a single processor workstation.

In order to determine the size of circuits in number of transistors for which there is speedup in simulation on a cluster with two nodes, parallel simulations using the presented algorithm were performed on circuits consisting of various number of MOSFETs. These circuits are generated by successive replication of bilinear SC filter circuit with MOSFET operational amplifiers. Then speedup is calculated acording to (1).

The simulations were performed on the Beowulf cluster whose structure is given in Table 1. The generated simulation results are given in Table 2.

Component	**Specification**
Master node	PC Pentium IV, 2.4GHz, 1GB RAM, 240GB HDD
Slave nodes	8× PC Pentium IV, 2.4GHz, 512MB RAM, 80GB HDD
LAN	1Gbit Ethernet

Table 1. Beowulf cluster structure

Number of MOS-FETs	**Simulation Speedup (2 cluster nodes)**
420	0.96
740	1.1
1480	1.5

Table 2. Speedup of parallel simulation in pAlecsis

As it is shown, parallel simulation time almost equals sequential simulation time for circuits containing 420 MOSFETs (speedup is approximately 1). For such and bigger circuits simulation time on 1 node is bigger than simulation time on 2 nodes, so the parallel simulator gives speedup in simulation (see Table 2).

References

[1] Sterling, T., *Beowulf Cluster Computing with Linux*, MIT Press, 2001.

[2] Gropp, W., Lusk, E., and Skjellum, A., *Using MPI: Portable Parallel programming with the Message-Passing Interface*, second edition, MIT Press, 1999.

[3] Gropp, W., Lusk, E., and Thakur, R., *Using MPI-2: Advanced Features of the Message-Passing Interface*, MIT Press, 1999.

[4] http://www.cs.sandia.gov/xyce/

[5] Fröhlich, N., Riess, B. M., Wever, U., Zheng, Q., *A New Approach for Parallel Simulation of VLSI-Circuits on a Transistor Level*, IEEE Transactions on Circuits and Systems, Part I, Proc. Int. Conference on Parallel and Distributed Processing Techniques and Applications, pp. 601-613, Vol. 45, No. 6, June 1998.

[6] Martin, D. E., Radhakrishnan, R., Rao, D., Chetlur, M., Subramani, K., Wilsey, P., *Analysis and Simulation of Mixed-Technology VLSI Systems*, Journal of parallel and distributed computing, vol. 62, No 3, pp. 468-493, 2002.

[7] Savić, M., Anđelković, B., Litovski, V., *Parallel Mixed-Mode Simulation – Preliminary Study*, Proc. INDEL 2004, Banja Luka, pp. 76-79, 2004.

[8] Litovski, V., Zwolinski, M., *VLSI Circuit Simulation and Optimization*, Chapman and Hall, London, 1997.

[9] Mrčarica, Ž., et al., *Alecsis 2.3, the simulator for circuits and systems. User's Manual*, Laboratory for Electronic Design Automation, Faculty of Electronic Engineering, University of Niš, Yugoslavia, LEDA-1/1998, http://leda.elfak.ni.ac.yu /projects/Alecsis/alecsis.htm

[10] Litovski, V., Maksimović, D., Mrčarica, Ž., *Mixed-Signal Modeling with AleC++: Specific Features of the HDL*, Simulation Practice and Theory 8, pp. 433-449, 2001.

[11] Mrčarica, Ž., Ilić, T., Glozić, D., Litovski, V., Detter, H., *Mechatronic Simulation Using Alecsis: Anatomy of the Simulator*, Proc. Eurosim'95, Vienna, Austria, pp. 651-656, 1995.

[12] SuperLU, http://crd.lbl.gov/~xiaoye/SuperLU/

Corresponding Author: Bojan Anđelković,
Laboratory for Electronic Design Automation,
Faculty of Electronic Engineering,
Aleksandra Medvedeva 14, 18000 Niš, Serbia

Received: September 18, 2007
Revised: October 15, 2007
Accepted: October 25, 2007

A Petri Net approach to ARGESIM Benchmark C2 'Flexible Assembly System' using the MATLAB PetriSimM toolbox

Thomas Löscher, Christoph Habersohn, Gregor Hinker, TU Wien, Austria
thomas@loescher.at, *christoph.habersohn@gmail.com, gregor.hinker@esperanto.at*

Simulator: The sophisticated MATLAB PetriSimM toolbox offers the capabilities of analysis, modelling and simulation of Petri Nets. Furthermore it is possible to optimise scheduling problems based on Timed, Coloured, and Stochastic Petri Nets. The open source MATLAB PetriSimM toolbox can be used for education in a graduate level and for modelling and simulating real life processes of discrete event systems in equal measure. The toolbox is embedded in the MATLAB environment and its usage requires version 7.0 or higher. In this toolbox a sophisticated way of defining firing sequences and priorities is implemented. Furthermore, results can be shown with Gantt charts and marking plots or the data can be exported for external post-processing.

Model: An assembly system described in Comparison 2 is an atypical application for Petri Nets. The aim of this simulation was to prove that even such a process can be simulated by a discrete system based on Petri Nets, although the programming and processing efforts are considerably high. The C2 comparison "Flexible Assembly System" consists of two times two conveyors, eight assembly stations and two shifting parts.

The priorities at the switches in front of each assembly station have been assigned as follows:

1. Leaving the assembly station
2. Entering the assembly station
3. Forwarding along the conveyor

At the end of the conveyor, the shifting part obeys similar rules:

1. Shift directly from station to station
2. Shift from station to conveyor
3. Shift from conveyor to station
4. Shift from conveyor to conveyor

All these prioritizations can be represented as conflicts of the Petri Net. Therefore, priorities are assigned to selected transitions to solve these control strategies by the use of the basic properties of Petri Nets.

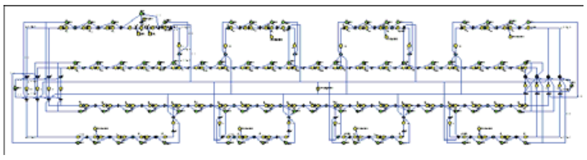


Figure 2. Whole Model

In contrast to most usual simulation strategies the palets are not modelled as places, but as combination of multible coloured tokens, the colours making statements about the state of procession of the palet. Therefore, following one single defined palet through the itinerary is not evidently possible and post-procession of the node data becomes necessary.

Figure 1 shows a screenshot of the whole model. Each procession unit and the discretized conveyor elements are designed using three-places-units (see figure 2). A "redlight" place ("frei") signalizes whether entrance into the unit is possible or not. Duing the given processing time the tokens pass from the entranceplace to the exitplace. For a more concise data recording, the processing unit additionally disposes of a place ("Bearbeitet") indicating when the palet is assembled.

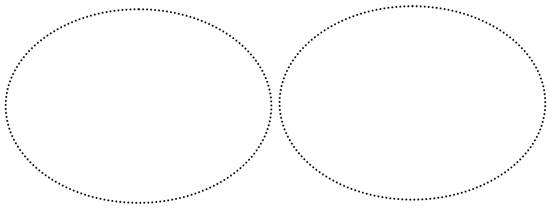


Figure 1. Three-places-units: conveyor unit and processing unit

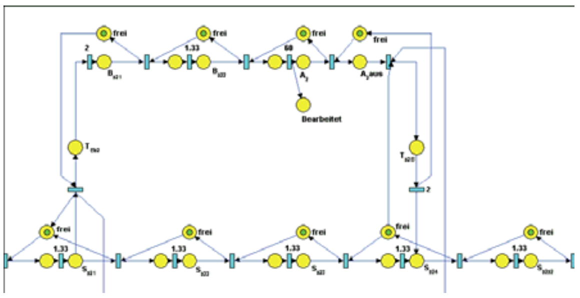


Figure 3. An assembly station

An assembly station (Figure 3) consists of successive conveyor units and one procession unit, both mentioned above. In order to enable simulations with a number of palets exceeding the band capacity (40), an additional tool had to be introduced: this control unit permits loading of new palets from the assembly stations to the main conveyor only in case of free capacity and thus avoids system deadlocks.

The complexity of the model (243 places, 8 colors) results in largescaled matrices (from 1145 x 25.670 to 1145 x 612.434) that increase both processing time and quantity of data: each single simulation took between one and three days on a home computer; furthermore, up to 29 GB of data were produced during one simulation session, in total 110 GB. Processing was therefore delegated to the phoenix-cluster of the TU Wien.

A-Task: Due to systematic restrictions of the method and the model it is not possible to follow one single palet throughout the process; hence, the statistical evaluation was done by external post-processing of the logfiles of every place node based on small C programs. Following the recommendation, the data used was from the 120th to the 600th minute of simulation.

B-Task: The simulation time was eight hours.

Number of palets	Total througput	Average throughput time [s]
5	480	292.50
10	960	292.50
13	1333	273.29
14	**1440**	**272.50**
15	1440	292.50
20	1444	391.92
40	1441	792.04
60	1433	1198.66

Table 1. System behaviour: through put time (extract)

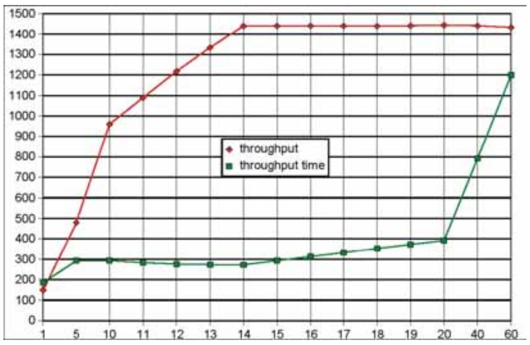


Figure 4. System behaviour: throughput time/throughput

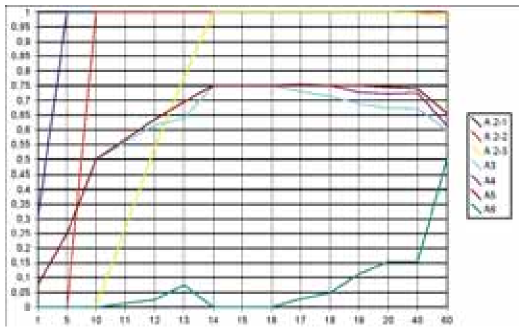


Figure 5. System behaviour: assembly stations' utilization

Table 1 show an extract of the throughput times and the graph of Figure 4 aims to provide an overview of the system's behaviour:

Larger calculations than with 40 palets are purely theoretic, since they exceed the system's capacity. This was made possible by the deadlock control unit.

C-Task: As shown in the following graph (figure 5), the utilization of stations A2–A6 reaches its possible maximum (100% utilization of "bottleneck" stations A2-1–A2-3) starting at 14 palets. In this configuration, the highest possible throughput (1440 palets; at some points slightly higher numbers may have encountered due to the delayed measurement) can be obtained at the lowest throughput time (272.50 seconds).

Resumé: This work shows an alternative modelling approach using Petri nets and the sophisticated PetriSimM toolbox.

Corresponding author: Thomas Löscher,
Department of Analysis and Scientific Computing
Vienna University of Technology,
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
thomas@loescher.at

Received: June 12, 2007
Revised: October 10, 2007
Accepted: October 20, 2007

An Object-Oriented Solution to ARGESIM Benchmark C2 'Flexible Assembly System' using TaylorED

Thomas Löscher, Yilin Huang, Voin Legourski, Ondrej Cevan, TU Wien, Austria
thomas@loescher.at, yilin@gmx.net, legourski@yahoo.com, ondrejcevan@yahoo.com

Simulator: Taylor Enterprise Dynamics (TaylorED) is an object-oriented comprehensive tool for modelling, simulating, visualizing, and monitoring dynamic-flow systems. Atoms (comparable to classes in object-oriented programming languages) are TaylorED's reusable "smart objects" that inherit behaviours from their parent atoms. TaylorED's standard atom library offers over 170 predefined atoms (new atoms are available monthly to download). Users can create and customize their own atoms and building-blocks with "four dimensions": location, speed, rotation and dynamic behaviour over time. Models can be viewed dynamically and simultaneously in 2D, 3D, and Virtual-Reality animations. With "Atom Concept", the soft-ware is powerful, flexible, and easy-to-use for model building and simulation analysis. Simulation results may also be exported to external software programs.

Model: The Flexible Assembly System model is composed of eight almost identical submodels. Each of them contains seven standard atoms as shown in Figure 1.

Figure 1: Screenshot of a submodel

The assembly-station (A1), shifter1 and shifter2 are represented by Server atoms, which have attributes Cycletime among others to configure the durations that pallets being processed by the Server.
The conveyor belt (in dark green) leading pallets to the assembly-station is represented by an Accumulating Conveyor atom. The belt leading pallets away from the assembly-station and the one direct connecting shifter1 and shifter2 are represented by Non-Accumulation Conveyor atoms. The length, speed, capacity, etc. of a conveyor belt can be configured. A Queue atom is placed after shifter2 to perform priority checking of pallets from the two different belts. Additionally, the pallets being processed in the system are Product atoms. Loading the pallets into the system and unloading of them are realized by Source and Sink atoms. Lock and Unlock atoms are responsible for controlling the amount of pallets circulating in the system.

More advanced strategy controls, such as input/ouput channels, triggering-events, label-settings, exporting time-stamps to Excel files, as well as other simulation constrains can be defined by using 4Dscript, which will be described as following.

Shifter1 decides whether it shall shift a pallet towards an assembly-station through the upper belt (by output channel 1), or else directly shift the pallet to shifter2 through the lower belt (by output channel 2). The decision is made according to a pallet's label value (which shows if the pallet has been in the station) and to the fact if the belt (in dark green) towards the station is full. The 4Dscript of shifter1, e.g. in front of station A2, is listed below.

```
if(label([step1], rank(1,c)) = 2, 2,
if(or(label([step2], rank(1,c)) = 0,
      label([step2], rank(1,c)) = 12),
if(content(out(1,c)) =
   att([Capacity], out(1,c)), 2, 1), 2))
```

Each pallet will get an updated label-value after being processed and before leaving an assembly-station. The 4Dscript of setting labels, e.g. in station 5, is shown below. Station 3, 4 and 5 will add a value of 3, 4 or 5 correspondingly to label step2. Station 2 will have the label step1 value set to 2.

```
setlabel([step2],label([step2],i)+5,i)
```

The pallets' colors will be changed after each assembly processing for visualization purposes. For example, if all steps of station 2, 3, 4 and 5 have been completed, the icon of a pallet will be set to a green circle.

```
if(
 Label([step1],i)=2,
 if(
  label([step2],i)=0,
  set(icon(i),27),
  if(
   label([step2],i)<6,
   set(icon(i),29),
    set(icon(i),24)
   )
  ),
  if(
   label([step2],i)=0,
   set(icon(i),47),
   if(
    label([step2],i)<6,
    set(icon(i),50),
    set(icon(i),48)
   )
  )
 )
```

A-Task: The timestamps that each pallet entering and exiting the assembly-system are exported to an Excel file for statistic purpose. Entering time is recorded by Source atom using

```
ExcelWrite(output(c), 1, time),
```

whereas entering time is recorded by Sink atom using

```
ExcelWrite(value(name(i)), 2, time).
```

B-Task: The system model was simulated for eight hours with different amounts of pallets (from 15 ro 60) circulating in it. The throughputs were counted by the *Monitor* facility of the simulation tool. The throughput per hour can be summed dynamically be the tool as well. The throughput time of each pallet was calculated by Excel.

	Through-put	**Average Throughput time**	**Min Throughput time**	**Max Throughput time**
15	1255	343.98	213.67	406.33
16	1365	337.30	213.67	407.33
17	1440	339.79	240.00	400.00
18	1440	339.71	240.00	400.00
19	1440	380.00	340.00	400.00
20	1440	399.96	340.00	450.00
40	1440	790.77	524.00	17730.00
60	1440	1133.78	402.00	5740.00

Table 1: Throughput and throughput time of pallets

C-Task: The data in Table 1 shows that the increased amount of pallets in the system doesn't necessarily result in an improved performance; on the contrary, it may downgrade the performance. With 17 pallets on the conveyors, the throughput already reaches its peak, and the throughput time is considerably efficient. Above the number of 17, the throughputs stagnate at 1440, and the throughput time start to rise gradually. Specially when there're more than 20 pallets in the system, the level of throughput time surge dramatically, as the pallets remained substantially longer in the system because of the overcrowded traffic.

Figure 2: Throughput of 20 ~ 60 Pallets

Figure 3: Throughput Time of 20 ~ 60 Pallets

Resumé: This work shows a classical object oriented modelling approach using Taylor Enterprise Dynamics. The statistical evaluation is done by the use of an external Excel file where the shown fgures could be easily created.

Corresponding author: Thomas Löscher,
Department of Analysis and Scientific Computing
Vienna University of Technology,
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
thomas@loescher.at

Received: June 20, 2007
Revised: October 10, 2007
Accepted: October 20, 2007

A System Dynamics Modelling Approach to ARGESIM Benchmark C3 'Class E Amplifier' using AnyLogic

Johannes Asamer, arsenal research, Austria, *johannes.asamer@arsenal.ac.at*
Štefan Emrich, Vienna Univ. of Technology, Austria, *semrich@aurora.anum.tuwien.ac.at*

Simulator: AnyLogic is an object-oriented dynamic simulation tool which brings together System Dynamics, Process-centric and Agent Based approaches within one modeling language and one model development environment. AnyLogic contains a set of simple objects beside library objects and allows to model different tasks from several areas. These elements can be added to a project simply by drag and drop, and parameterized afterwards. Moreover it is possible to include java code into some elements, which can be executed on demand.

Model: The simulation of a class E amplifier, first introduced by N.O. Sokal and A.D. Sokal is described in this work. By applying Kirchhoff's circuit laws following equations are obtained:

$$dx_1 / dt = (-x_2 + VDC) / L1 \tag{1}$$

$$dx_2 / dt = (x_1 - x_2 / R(t) - x_3) / C2 \tag{2}$$

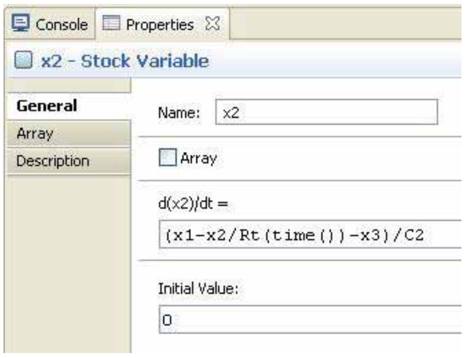
$$dx_3 / dt = (x_2 - RL * x_3 - x_4) / L3 \tag{3}$$

$$dx_4 / dt = x_3 / C4 \tag{4}$$

x_1, x_2, x_3 and x_4 represent currents and voltages in the circuit. For the simulation of the class E amplifier following elements are used: Parameters Stock, Variables, Table Functions, Flow/Auxiliary Variable. In our case the parameters represent the constant properties of electronic components like resistance, inductivity, capacity and direct voltage sources. After adding these elements the according values have to be defined.

For the definition of differential equations AnyLogic makes use of System Dynamics approach: stock variables represent states, and after the definition of a name, the derivation of it (the flow) can be used in an equation (see Figure 1). The used equation contains a parameter (C2), other stock variables (x_1, x_2, x_3) and a table function (R(t)) with the predefined function time() as argument. The circuit can be described with four differential equations and therefore also four stock variables are used (x_1, x_2, x_3, x_4). Additionally an initial value for the according stock variable has to be set.

AnyLogic uses arrows to visualize the connections of



x2 - Stock Variable

Name: x2

Array

d(x2)/dt =

```
(x1-x2/Rt(time())-x3)/C2
```

Initial Value:

```
0
```

Figure 1: Syntax of AnyLogic, Equation (2)

parameters and variables depending on the input equations. An arrow from element A to element B means that the output of A is used in B (see Fig. 2).

As AnyLogic offers no signal sources we used the function 'time()' in combination with a table-function. Defined by a look-up table such a function returns an output-value depending on the input. The values in the table represent the highs and lows together with the corresponding time, values between the given times are interpolated linearly. Since time is continuously growing it will eventually exceed the period of time for which R(*t*) has been defined, in our case 10E-6 seconds. Fortunately the table function offers a feature called 'repeating' (modulo - function.

For visualization of the voltage on the load a flow variable has to be added. Afterwards a formula was defined as for the definition of the stock variables (compare Fig. 1), which writes the result to the defined flow variable. In our case this formula is simply the product of current x_3 and load resistance.

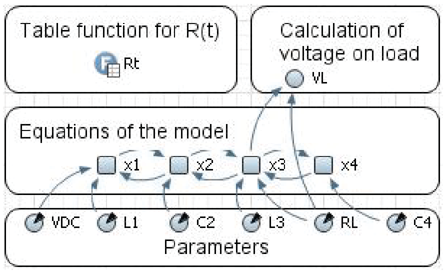


Figure 2: AnyLogic model

A-Task: A possibility to calculate the eigenvalues of the system could not be found in AnyLogic. Determination of the eigenvalues would be necessary to determine the characteristics of the system and thus to choose an adequate ODE-solver for the model. But since AnyLogic does not offer an option for choosing a solver either, this task is redundant.

B-Task: The execution of the model is started by pressing the 'Run-Button'. Upon this any occurring errors are displayed in an extra window. The visualization of the currents and voltages of the model is done by using datasets, which store the values of variables during runtime. The sampling-interval as well as the number of points which have to be stored is adjustable in the properties of the dataset element. For the visualization it is further possible to plot datasets against time as well as against another. The quality of the visualization is dependent on the sampling interval. If it is too small the visualized curve will look rough or even worse, essential effects may be masked since the points are only connected by a straight line. Figure 3 visualizes the results of the simulation.

The output looks very similar to the results of other simulations (e.g. Matlab/Simulink) and is compatible to the theories of a class E amplifier.

Generally the simulation proceeds in real-time (what is visible on the output of the time function) and can be accelerated or even slowed down. Because of the high frequency signals in this model a real-time calculation is not possible (on an Intel Dual Core 1.88GHz processor). It took about 5 seconds to compute a period of 1 millisecond within the model. But since the period of the output is only 0.01 milliseconds and the transient effect takes not more than three periods, this is sufficiently fast.

C-Task: The purpose of task C is to analyze the impact of different rise/fall times (TRF) on R(*t*). To investigate four different TRFs in one simulation run, four separate models have to be created, although they only differ in the definition of the table function. As visible in table 1, in row two and four the TRF is the argument for the function respectively it is added to the half-period. In figure 4 the TRF is varied between 1E-7 and 1E-15.

The results are again equal to those produced with other simulation tools (Matlab/Simulink). Only the course of the phase for a TRF of 1e-7 does slightly differ.

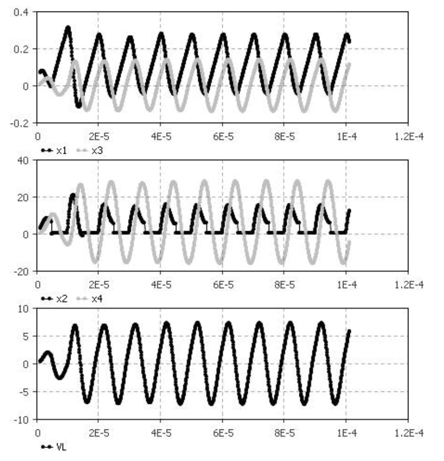


Figure 3. Simulation results

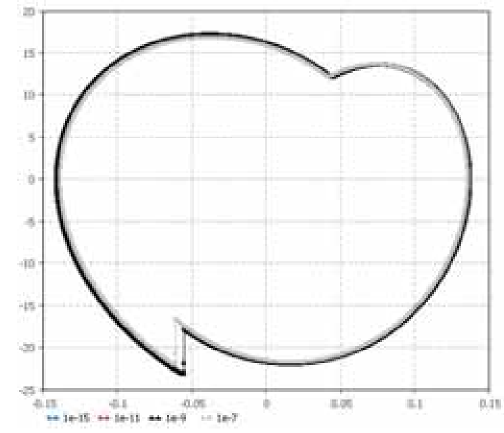


Figure 4: Phase diagram for different TRF

Resumé: As we went to create a model of the E amplifier defined by ARGESIM comparison 3 in AnyLogic, we were able to notice several interesting aspects. First of all we had to find a way to realize time-dependent variables. When trying to solve the given problem it became evident that AnyLogic is lacking the possibility to directly compute eigenvalues of the investigated system. Since this can be bypassed by the use of other (mathematic) software it does not pose a too large shortcoming. Nevertheless the lacking possibility to choose an ODE-solver does appear to be quite a problematic shortcoming. But despite this shortcoming the computed results are apparently correct.

Corresponding author: Johannes Asamer,
arsenal research, Transport technologies,
Giefinggasse 3, 1210 Vienna, Austria
Johannes.asamer@arsenal.ac.at

Received: September 12, 2007
Revised: November 10, 2007
Accepted: November 18, 2007

Comparison of Circuit Diagram Modelling and ODE Modelling for ARGESIM Benchmark C3 'Class-E Amplifier' using Dymola

Günther Zauner, Phillip Jahn, Thomas Polzer, Alexandra Schuster, Vienna Univ. of Techn., Austria

Simulator: Dymola is a multi domain simulation tool based on the Modelica standard for complex dynamic systems. It enables the user to modify existing or create custom libraries especially for physical modelling.

Since the simulator includes already a wide range of predefined libraries, Dymola can be used in various areas of engineering including electronics and mechanics.

The models can be created graphically (using real world elements as resistors or capacitors) with ideal components or also with "real world" elements or text based. Furthermore it is possible to script the simulation, so a re-simulation can be done very easily.

Model: A class-E power amplifier circuit was modelled. Basically it is an amplifier circuit with a switching resistor (R). Its schematic can be found in Figure 1.

The resistance-time diagram of the switching resistor R can be found in Figure 2. It enables the circuit to generate an AC signal out of a single DC voltage source. A combination of capacitors and inductors are used to smooth the output signal. Therefore, instead of a rectangular signal, a nearly perfect sine signal is produced.

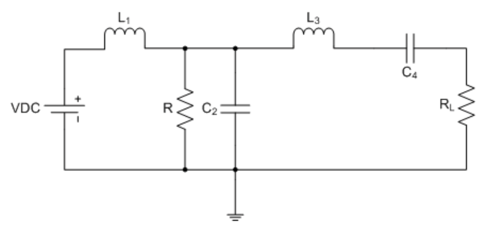


Figure 1. Class-E Amplifier circuit with time-dependent resistance *R*

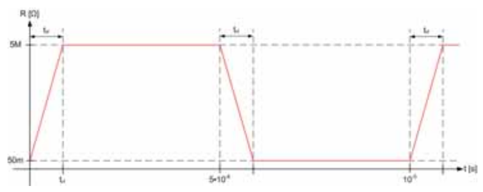


Figure 2. Resistance-time diagram of switching resistor R. In the "on"-period the resistance is only 50 mΩ, hence nearly a short-circuit.

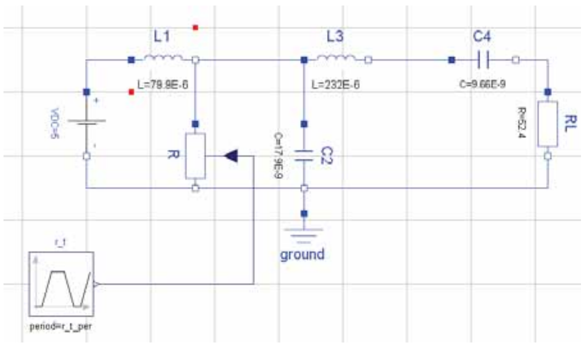


Figure 3. Dymola Circuit Model defined in the *Diagram* layer with components from the Modelica standard library

We have decided to model the circuit in two different ways. The first solution is modelled directly as circuit using the predefined electronic components. Furthermore, we have textually modelled the system using its differential equations.

In the first case, the circuit was directly input into the modelling system. The resistance-time diagram for the switching resistor was modelled as a *trapezoid* signal, from `Modelica.Blocks.Sources`. The model can be found in Figure 3.

In the second case we have written a text based model description. This description contains the parameters of the circuit, its system matrices (with switched on and off resistor), and the state variables of the system. Additionally the eigenvalues are calculated by using the function `eigenValues` in the `Modelica.Math.Matrices` library.

In the equation section the differential equations are defined. The model can be found in Listing 1.

```
model class_e_amplifier_dgl
  // Curcuit parameters
  parameter Real u = 5.00E+00;
...
  // Resistance diagram of R
  Modelica.Blocks.Sources.Trapezoid r_t(
    period = r_t_per,
    offset = r_t_on,
...
  // System matrix (in ON and OFF state)
  parameter Real system_matrix_on[4,4] =
    [0,    -1/l1,          0,      0;
     1/c2, -1/(r_t_on*c2), -1/c2,  0;
```

SNE 17/3-4, December 2007

```
      0,    1/l3,           -rl/l3, -1/l3;
      0,    0,              1/c4,   0];
...
  // State variables of the curcuit
  Real x1(start = 0), x2(start = 0),
       x3(start = 0), x4(start = 0);
  // Additional interesting signals
  Real y, v_l3, i_r;
  // Calculation of the eigenvalues
  parameter Real eigen_values_on[4,2] =
    Modelica.Math.Matrices.
        eigenValues(system_matrix_on);
  parameter Real eigen_values_off[4,2] =
    Modelica.Math.Matrices.
       eigenValues(system_matrix_off);
equation
  // System differential equations
...
  // additional signals
  y = rl*x3;i_r = x2/r_t.y;
  v_l3 = der(x3)*l3;
end class_e_amplifier_dgl;
```

Listing 1: Differential Equation model of the class – E amplifier as defined in ARGESIM Benchmark 3

The whole simulation process was scripted using the built-in function of Dymola. Therefore it is possible to resituate and view the results with just one mouse click.

Note: Task A was only done using the second model. Task B and C were executed on both models.

A-Task – Calculation of the eigenvalues: The resulting eigenvalues are:

On period	**Off period**
−1,1303e+005 + 6,5835e+005i	−5,4708 e+004 + 1,0407e+006i
−1,1303e+005 – 6,5835e+005i	−5,4708 e+004 - 1,0407e+006i
−6,258e+002	−5,8228e+004 + 5,3275e+005i
−1, 117e+109	−5,8228e+004 - 5,3275e+005i

B-Task: In this task we have simulated the initialization of the circuit in the time interval of 0 to 100 µs. As integration algorithm Dassl was used, because it offers adaptive step size. The tolerance was set to 10^{-5}. The resulting voltage-time diagram can be found in Figure 4.

Furthermore the values of the state variables at the end of the simulation were exported into a script file. This script is used in Task C to initialize the circuit.

C-Task: In this task a parameter study was done. Therefore the rising and falling time of the switch resistor R were variated. The used values are: $1\text{fs}, 10\text{ps}, 1\text{ns}, 1\mu\text{s}, 10\mu\text{s}$.

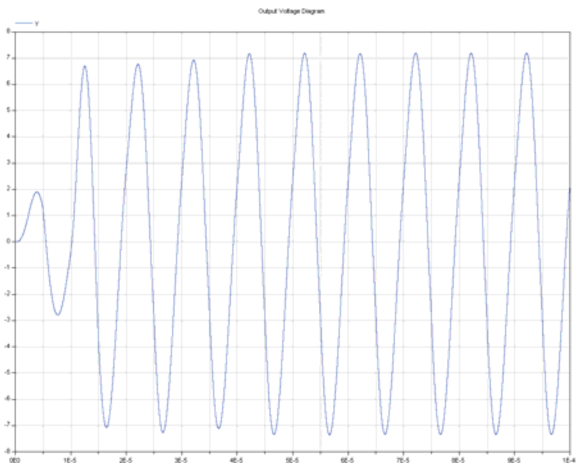


Figure 4. Output voltage at the load-resistance

The same simulation settings as in Task B were used. Only the simulation interval was changed to [100 µs, 109 µs]. The differences in the output voltage as well as the resulting voltage-current diagrams of inductivity L_3 were analyzed (Figure 5).

Resumé: The tasks have been done in textual mode and with electrical modelling Modelica standard library. The eigenvalue calculation has been done by using a Modelica function.

Using script files affords easy parameter variation and rollback of simulation tasks with predefined solver parameters.

Corresponding author: Günther Zauner,
Vienna University of Technology
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
guenther.zauner@drahtwarenhandlung.at

Accepted EOOLT 2007, June 2007
Received: August 10, 2007
Accepted: August 20, 2007

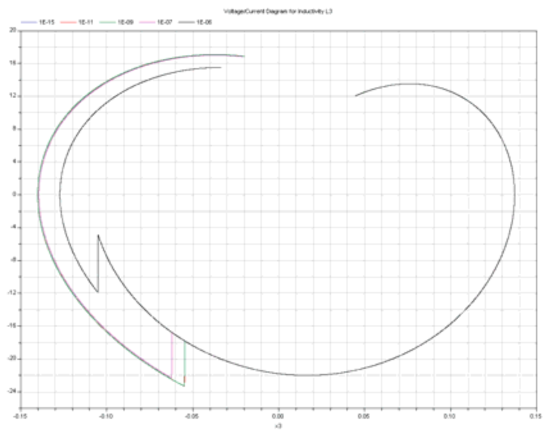


Figure 5. Voltage-Current Diagram of L3

Simulation and Analysis of ARGESIM Benchmark C3 'Class-E Amplifier' based on a Simulink Model with MATLAB functions

Johannes Asamer, arsenal research, Austria
Günther Zauner, Vienna University of Technology, Austria

Simulator: MatLab is platform independent software, developed by MathWorks. The software was originally designed for matrix calculations (Matrix Laboratory). Programs can be written as executable script, which allows a generation of (commercial) toolboxes for general use. Code can also be exported to other programming languages (e.g. C or Java).

Simulink also has been developed by MathWorks and needs MatLab for execution. By using predefined blocks different tasks can be modelled graphically. Moreover it is possible to create new blocks by implementing MatLab-Code. Additional blocks with complex functions from many areas of science are available. With Simulink, dynamic systems can be modelled and it is used in control theory and signal processing. Because it is based on MatLab the solutions are also calculated numerical, wherefore several ODE-solver can be used.

Model: In following section the modelling of a class E amplifier, introduced by N.O. Sokal and A.D. Sokal is described. By applying Kirchhoff's circuit laws the amplifier can be characterized by following formulas:

$$dx1/dt = (-x2 + VDC)/L1 \tag{1}$$

$$dx2/dt = (x1 - x2/R(t) - x3)/C2 \tag{2}$$

$$dx3/dt = (x2 - RL * x3 - x4)/L3 \tag{3}$$

$$dx4/dt = x3/C4 \tag{4}$$

The aim was to implement these characteristic formulas into MatLab/Simulink. Therefore four blocks have been created, which contain the formulas. The properties of one block can be defined by referencing to a MatLab-script, which contains the associated formula. The input of each block is a vector with four elements ($\vec{x}$) and the output is a single value, representing the first derivation of one element of $\vec{x}$. Although inside the formulas just some elements of the vector are used, the whole vector is transfered. A single element is accessed just by declaring the according index of the vector-element.

The definition of formula 1 in the MatLab script is described in the following way:

```
function dx1 = equn1(x)
global VDC RL L1 C2 L3 C4
dx1 = (-x(2) + VDC)/L1;
```

By applying an integrator to the output of each block, the original (neither integrated nor differentiated) values are obtained. Afterwards these values are put together to one vector by using a multiplexer, provided by Simulink. Actually the multiplexer is used to put different signals on one bus, but is interpreted in this project as a vector with four elements. Moreover for the input of formula 2 a fifth element has to be added to the vector (or bus) namely the controlling resistance $R(t)$. Because Simulink is a common environment for modelling signal flow, there are different kinds of signal sources. One of them is called 'repeating sequence' and is here used for modelling $R(t)$. For the definition of this block a set of paired values is necessary. The first value of a pair indicates a point of time and the second value the output value for this point. Values between two points are interpolated linearly. Because the signal form of $R(t)$ is a rectangle with a rise and fall time (TRF) five points have to be defined.

An alternative would be to build up the whole system with blocks and use one line for each signal. The advantage of the chosen way to model the Class-E amplifier is the compact structure. By using the blocks which are able to execute a predefined MatLab code, the big part of the model is reduced to a set of four blocks. Moreover the usage of signal busses leads to more clearness in representation of the model.

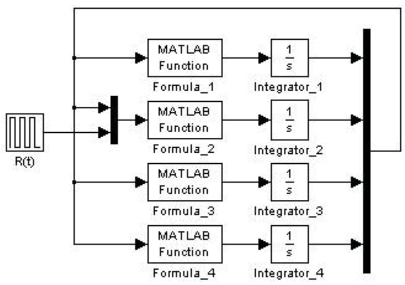


Figure 1. Simulink block diagram

A-Task: The first task is to calculate the eigenvalues of the system in the ON- and OFF period of $R(t)$. To do so first the Jacobi-matrix has to be calculated, which is the derivation of the model formulas with respect to $\vec{x}$. As mentioned above $\vec{x}$ contains four elements, which represent the currents and voltages in the circuit.

Because the element in the second row and second column of the matrix depends on $R(t)$, the Jacobi-matrix changes over time. MatLab contains a function called $eig()$, which is suitable for calculating the eigenvalues of a matrix and therefore has been applied for this task.

The result of $eig(J\_on)$ and $eig(J\_off)$:

$$J_{OFF} = \begin{bmatrix} -54708 + i(1.04e+6) \\ -54708 - i(1.04e+6) \\ -58228 + i(5.33e+5) \\ -58228 - i(5.33e+5) \end{bmatrix} \quad J_{ON} = \begin{bmatrix} -1.12e+9 \\ -625.78 \\ -113040 + i(6.59e+5) \\ -113040 - i(6.59e+5) \end{bmatrix}$$

Out of the eigenvalues different conclusions can be drawn concerning the system behaviour. All eigenvalues have negative real parts which mean that the system is stable. For the OFF-Period the system can be observed as not-stiff because the absolute values of the eigenvalues are approximately in the same order of magnitude, in contrast to the eigenvalues of the ON-period, where this is not the case. This means the system behaviour changes between stiff (ON) and not stiff (OFF).

B-Task: The circuit has been build up in MatLab/Simulink as described above. Before executing the simulation an appropriate ODE-solver has to be chosen. Basically it has to be decided between solver with variable and fixed step size. Because in parts the system is stiff a solver with variable step size has to be used. It is also possible to use this solver for none-stiff systems but it would be problematic in terms of calculation time to use a solver with fixed step-size for stiff systems.

MatLab/Simulink offers seven different ODE-solvers with variable step size for continuous simulation. All of them resulted in (approximately) the same output.

Beside the runtime, differences could be observed in the form of the current and voltage on the output resistance. Basically the current on the load resistanceshould be zero except the periodically repeating peaks. But for three ODE-solvers (ODE45, ODE23 and ODE113) a noise with quite big amplitude (~ 2V)

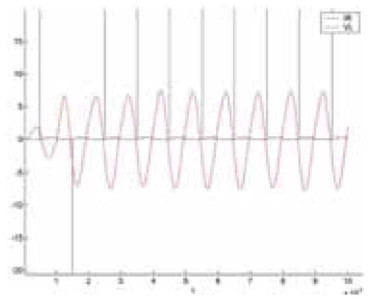


Figure 2. Desired output calculated with ODE23s and ODE23tb

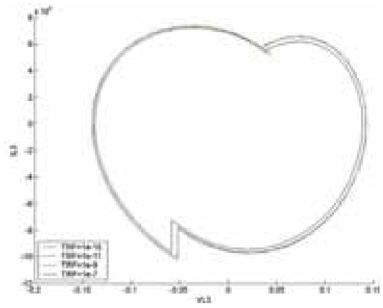


Figure 3. Phase diagram of $VL3$ versus $IL3$

was overlaid to the current. For two other solvers (ODE15s and ODE23t) the voltage on the output showed a blurred form. Only two of the seven solvers (ODE23s and ODE23tb) provide the results which were expected from the simulation.

C-Task: In the third task the influence of the rise and fall time of $R(t)$ (TRF) was investigated. Therefore a phase diagram was visualized containing the current and the voltage on coil $L3$. For the different simulation runs the TRF has been varied between 100ns and 1fs. Again the results are very similar. Only at a TRF of 100ns the phase diagram looks slightly different.

Resumé: MATLAB/Simulink offers a convenient tool for modelling and analysis of stiff ODE – systems. As discussed above, the choice of the best suited ODE – solver for a defined problem is one of the main points to be focused on before starting a simulation run.

Corresponding author: Johannes Asamer,
arsenal research, Transport technologies,
Giefinggasse 3, 1210 Vienna, Austria
Johannes.asamer@arsenal.ac.at

Received: June 20, 2007
Revised: October 3, 2007
Accepted: October 21, 2007

Comparison of Classical IF-clause Modelling and Statechart Modelling for ARGESIM Benchmark C3 'Class-E amplifier' using MOSILAB/Modelica

Günther Zauner, Gemma Ferdinand Kaunang, Vienna Univ. of Technology, Austria;
gzauner@osiris.tuwien.ac.at

Simulator: Mosilab (Modeling and Simulation Laboratory) is a simulation tool for complex technical systems developed by Fraunhofer Gesellschaft. For the modeling process, MOSILAB uses the object- and equation-oriented model description language Modelica®, with a backwards-compatible extension to incorporate elements for describing model structure dynamics. This extension is defined in the model description language MOSILA. An integrated development environment offers users support in every work step – from model building over simulation to post-processing. Besides the traditional component diagram and textual modelling layer, class and statechart diagrams are available to users for the model-based development process.

MOSILAB can be used for applications in automotive and energetic systems, mechatronics, multi domain physics, block digram modelling and others.

Model: The basic class-E power amplifier was introduced by N.O. Sokal and A.D. Sokal in their classic paper from 1975. It is a switching-mode amplifier that operates with zero voltage and zero slope across the switch at switch turn-off. The equations describing the circuit are the state-equations where inductor currents and capacitor voltages are chosen as system variables. Kirchhoff laws for voltage and current deliver the following differential equations:

$$dx1/dt = (-x2 + VDC)/L1 \tag{1}$$

$$dx2/dt = (x1 - x2/R(t) - x3)/C2 \tag{2}$$

$$dx3/dt = (x2 - RL * x3 - x4)/L3 \tag{3}$$

$$dx4/dt = x3/C4 \tag{4}$$

The aim was to implement these equations into MOSILAB structure. Therefore two different ways have been chosen to model these equations.

The first solution is using textual Modelica notation. Designing the model is relativly easy in Modelica by using the exact equations above in the equation section and declaring all variables in the beginning section. The time dependent resistor $R(t)$ is modelled using an algorithm section, the whole modelling code is listed below:

```
model C3Mosilab_taskc
constant Real L1= 79.9E-6; C2= 17.9E-9;
...
Real x1(start=0.26144);
...
Real Rt; Real t_red; Real k;
Real IRT; Real VRL; Real derx3;
equation
    t_red= mod(time, 10E-6);
    k=((5e+6)-(5e-2))/TRF;
algorithm
   if (0<=t_red) and (t_red<TRF) then
      Rt:= (5e-2) + k*t_red;
   elseif(TRF<=t_red) and (t_red<(5e-6)) then
      Rt:= 5e+6;
   elseif ((5e-6)<=t_red)
              and (t_red<((5e-6)+TRF)) then
      Rt:= (5e+6) - k*(t_red - (5e-6));
   elseif ((5e-6)+TRF<=t_red)
              and (t_red<(10e-6)) then
      Rt:= 5e-2;
   else
      Rt:= -5;
   end if;
equation
   L1 * der(x1)= -x2 + VDC;
   C2 * der(x2)= x1 -(x2/Rt) - x3;
   ...
end C3Mosilab_taskc;
```

Listing 1: Mosilab Code in Modelica standard notation for the solution of Task C of the ARGESIM Benchmark 3

For the second solution the MOSILA language extension for statechart modelling is used, dividing the system in separated model parts, depending on the state of the time dependent resistor $R(t)$. It switches between the state $OFF(s1)$ and the state $ON(s2)$. Before simulation, $s1$ is set up as initial state. Thus, the model will change to $s2$ when the time dependent resistor $R(t)$ reaches value $50m\Omega$ and the model will again change to $s1$ when $R(t)$ reaches $5M\Omega$. By using the same code in an algorithm section as definied in the previous solution for $R(t)$, the value of the time dependent resistor is implemented. Declaring all variables in the beginning section and adding statechart code is realized as follows:

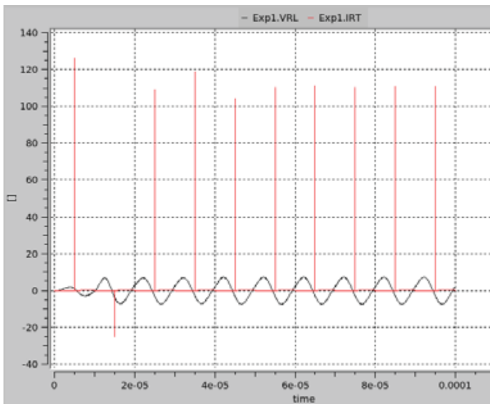


Figure 1: Time Curve $IR(t)$ and VRL

```
equation
   s1 = if Rt>=5e+6 then true else false;
   s2 = if Rt<=5e-2 then true else false;
statechart
state C3MosilabStateSC extends State;
  State State1; State State2;
  State Initial(isInitial=true);
  transition Initial->State1
    action Rs:=5e+6;
  end transition;
  transition State1->State2 event s2
    action Rs:= 5e-2;

  end transition;
  transition State2->State1 event s1
    action Rs:= 5e+6;
  end transition;
end C3MosilabStateSC;
```

Listing 2. Statechart code

The program codes of all solutions are done in the modelling section of Mosilab. Compiling the code, setting up the simulation parameter and simulation process is performed in the simulation section of Mosilab. The result of simulation is shown in the post-processing section.

A-Task: Calculating the eigenvalues of the system when $R(t)$ is $ON(50m\Omega)$ and when $R(t)$ is $OFF(5M\Omega)$ cannot be realized using Mosilab, because it does not have a `eigenValues` function in the Modelica library.

B-Task: Simulation of this task is done by setting `Dassl` as integration solver, $1e-10$ as min. stepsize, $1e-07$ as ma.x stepsize and $0\ldots1e-5\sec$ as simulation time interval. Choosing the initial value zero for $x1, x2, x3$ and $x4$, the results for current $IR(t)$ and output voltage VRL are presented in fig. 1.

C-Task: Simulation of this task was performed using the same settings as in task b apart from the simulation time interval which was chosen as $0\ldots9e-6\sec$. Reinitialization of the start parameters can be done using a script file or manually. Figure 2 shows the time curves for the voltage at coil $L3$ and the current at $L3$. Plotting this task is performed by using only a time curve of $VL3$ and $IL3$, because of lack of plot options for phase plane curves.

For all calculations and simulations, Mosilab 3.1 was used.

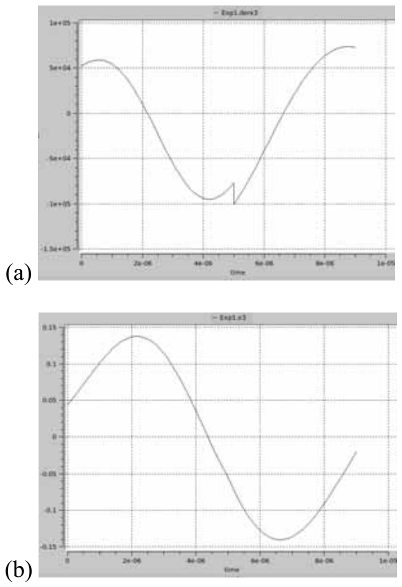


Figure 2: Time Curve (a) $VL3$ and (b) $IL3$

Resumé: Mosilab is a powerful simulation tool that makes it quite simple to model physical system, because it is based on Modelica, an object- and equation-oriented model description language. The unique part of Mosilab is that models can be implemented by using a statechart approach. This extension of the Modelica standard can be used to implement state events of any kind. A feature which can not be solved by every Modelica simulator or other standard simulation software. Mosilab has until now the disadvantage that it cannot calculate eigenvalues

Mosilab offers explicit as well as implicite algorithms for numerical integration and is therefore a adequate simulator.

Corresponding author: Günther Zauner,
dieDrahtwarenhandlung Simulation Services
Neustiftgasse 57-59, 1070 Vienna, Austria
guenther.zauner@drahtwarenhandlung.at

Received: October 6, 2007
Revised: October 20, 2007
Accepted: November 3, 2007

Comparison of Classical IF-Clause Modelling and State Chart Modelling for ARGESIM Benchmark C5 'Two State Model' in MOSILAB/Modelica

Gemma Ferdinand Kaunang, Günther Zauner, Vienna Univ. of Technology, Austria,

Simulator: Mosilab (**Mo**deling and **Si**mulation **Lab**oratory) is a simulation tool for complex technical systems. For the modeling process, MOSILAB uses the object- and equation-oriented model description language Modelica®, with a backwards-compatible extension to incorporate elements for describing model structure dynamics. An integrated development environment offers user support in every work step – from model building over simulation to post-processing. Besides the traditional component diagram, class and statechart diagrams are available to users for the model-based development process.

MOSILAB can be used for applications in automotive and energetic systems, mechatronics, as well as microsystems technology and others.

Model: This example tests the ability of the simulator to handle discontinuities of the type defined in ARGESIM benchmark 5. The problem is defined as follows

$$dy1/dt = c1 * (y2 + c2 - y1)$$
$$dy2/dt = c3 * (c4 - y2)$$

The model operates in two states:

1. the parameter $c2 = 0.4$ and $c4 = 5.5$ and it switches to state 2 if $y1 \geq 5.8$.
2. the parameters $c2$ and $c4$ change to $c2 = -0.3$ and $c4 = 2.73$. The model switches back to state 1 if $y1 \leq 2.5$.

The aim is to implement these equations and all the conditions above into Mosilab structure. Therefore two different ways are chosen to model these equations. The first solution is by using textual Modelica language. Designing the model is relatively easy by using the exact equations in the `equation` section and declaring all variables in the beginning section. The switching state is modelled in the `algorithm` section as follows:

```
algorithm
  when (y1>=5.8) then
     c2:=-0.3; c4:=2.73;
  end when;
  when (y1<=2.5) then
     c2:=0.4; c4:=5.5;
  end when;
```

Listing 1: Parameter event in a Modelica based `algorithm` section

The second solution is implemented by using the state chart approach, dividing the system into states, depending on the value of the variable $y1$. The method of modelling is quite similar to the first solution; namely, changing the switching state algorithm above to the statechart code as follows:

```
equation
  s2 = if y1 >= 5.8 then true else false;
  s1 = if y1 <=2.5 then true else false;
statechart
state C5MosilabStateSC extends State;
  State State1; State State2;
  State Initial (isInitial=true);
  transition Initial->State1
  end transition;
  transition State1->State2 event s2
    action
      c2:= -0.3; c4:= 2.73;
  end transition;
  transition State2->State1 event s1
    action
      c2:= 0.4; c4:= 5.5;
  end transition;
end C5MosilabStateSC;
```

Listing 2: State chart code

The program codes of all solutions are realized in the modelling section of Mosilab. Compiling the code, setting up the simulation parameters and simulation process are done in the simulation section. The results of the simulation run are shown in the post-processing section.

A-Task: The aim of task A is to simulate the system, by setting 1μsec as minimal stepsize, 80msec as maximal stepsize, relative tolerance of 1e-6 and `Dassl` as the integration method. Selecting the intial value 4.2 for variable $y1$ and 0.3 for variable $y2$, the result for value $y1$ is shown in Figure 1 for both model implementations. It took 0.1sec to simulate the task A for the classical Modelica solution and 0.3sec for the state chart solution.

Rel_Tol	1e-6	1e-10	1e-14
t_0	1.1088	1.1083	1.1090
t_1	2.1397	2.1394	2.1299
t_2	3.0588	3.0584	3.0592
t_3	4.0760	4.0757	4.0764
y1(5.0)	5.7988	5.7985	5.7997

Table 1: Time discontinuity and final value y1(5.0) with varying relative tolerance

solver	Dassl
t_0	1.1088
t_1	1.1220
...	...
t_{60}	4.9235
t_{61}	4.9370
y1(5.0)	5.7827

Table 2: Time series of state event and final value y1(5.0) with varying relative tolerance

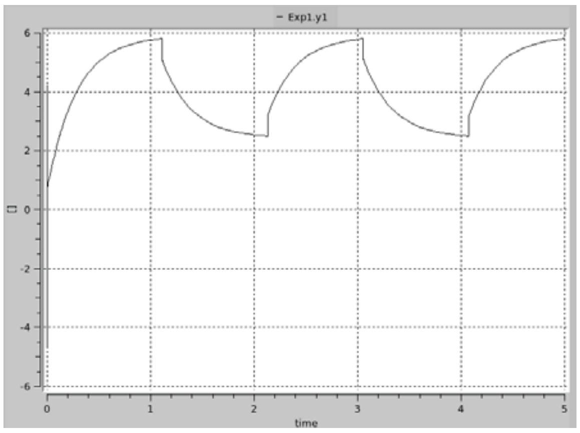


Figure 1. Time Curve y1

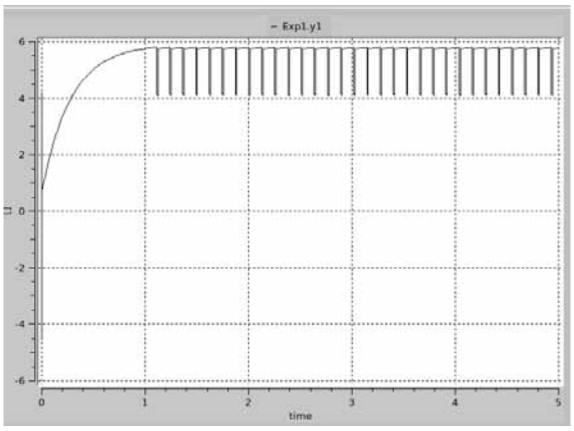


Figure 2: Time Curve of y1 for the settings of task D

B-Task: Based on Figure 1 the time discontinuity and the final value found can be seen at column two in Table 1.

C-Task: The parameter of relative tolerance is varied as follows: 1e-6, 1e-10 and 1e-14. Again, 1μsec is chosen as minimal stepsize. 80msec as maximal stepsize and `Dassl` as integration method. Table 1 shows the results of time discontinuity and final value y1(5.0) with varying relative tolerance.

D-Task: Changing the state 2 parameters c2 to -1.25, c4 to 4.33 and the switching condition to y1 $\leq$ 4.1 will result in a high frequent event of discontinuity for y1. Figure 2 shows y1 for both ways of implementation. The number of discontinuites found is 62.

The time discontinuities and the final values are shown in Table 2.

For all calculations Mosilab version 3.1 on Notebook Dell Latitude D630 Intel Centrino Duo was used.

Resumé: Mosilab is a simulation tool that offers several ways to model the system, because it is based on Modelica, an object- and equation-oriented model description standard. Furthermore the extension with state charts provides extra features for modelling structural dynamik systems.

The Dassl integration method implemented in Mosilab is an adequate state of the art solver for the given problem class.

Modelling the tasks in the textual mode gives faster simulation results than with state chart modelling. However the results remain the same.

Corresponding author: Günther Zauner
dieDrahtwarenhandlung Simulation Services
Neustiftgasse 57-59, 1070 Vienna, Austria
guenther.zauner@drahtwarenhandlung.at

Received: October 6, 2007
Revised: October 20, 2007
Accepted: November 3, 2007

A Directly Programmed Solution to ARGESIM Benchmark C8 'Canal-and-Lock System' using Visual Basic for Applications (VBA)

Thomas Löscher, Josef Mosser, TU Wien, Austria
thomas@loescher.at, e0126655@student.tuwien.ac.at

Simulator: Microsoft Excel is a spreadsheet application that provides simple to advanced means of creating and man-aging any type of list. To even make it more powerful and production oriented, MS Excel, like all members of the Microsoft Office family, ships with a computer language and a programming environment. Microsoft Visual Basic is one of the easiest programming languages to get started with for budding programmers. It has a very complete object model, easy to understand syntax (based around several flavours of traditional BASIC), and can provide Windows style interaction with the user. The Visual element in Visual Basic comes from the point and click interface - it is programmed visually, at least at the user interface level. Actions (scripts, macros) are attached to specific parts of the user interface. These are executed or evaluated at the time that the user interacts with the software. The general approach makes it very quick and easy to put together an application, and while the result might suffer from some performance issues, it is a great platform if performance is not the most important factor. Where VB comes into its own, however, is when it is integrated with other Microsoft applications. At some point, Microsoft introduced Visual Basic for Applications. This provided the power of the VB language coupled with the object model of the Office Suite. In other words, each Office component could be interacted with in a way that provided functionality beyond simple macro recording. Put another way; where users could record key presses, menu choices, and mouse movements and so on, now they could also write little programs to interact with the application in a conditional manner. In fact, VBA brings all the power of a programming language to the macro environment.

Model: The model represents a so called canal and lock system where barges move from east to west and vice versa. On their way through the channel the barges has to pass a lock where the water is raised and lowerd respectively if it is necessary. The used channel is very narrow and therefore barges can pass the channel only one after another in each direction. The defined system configuration leads to special control rules for the use of the channel and the lock.

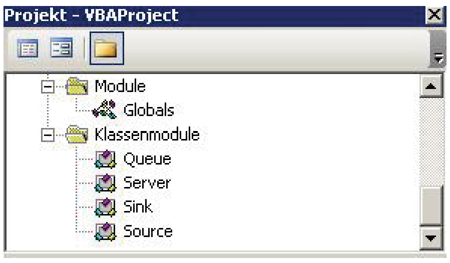


Figure 1. Class modules

A-Task: The model is implemented in an object-oriented way. Therefore the whole power of Visual Basic is used to define class modules for the needed objects. Figure 1 shows the four used classes: Source, Queue, Server and Sink. In these classes several properties are de-fined. An own event handler is programmed to model and simulate the behaviour of the barges, the lock and the channels. Macros are implemented to control the flow of the barges and to define and to change parameters and initial conditions.

For the input of the different parameters a simple user interface is created to give the user the ability to change the different input values of the Eastbound and Westbound barges. The produced simulation results are collected on special Excel spreadsheets and after the simulation Excel and VBA built-in functions are used to derive the mean, standard deviation and confidence intervals. Figure 2 shows a simple visualisation of the simulation model. The changing

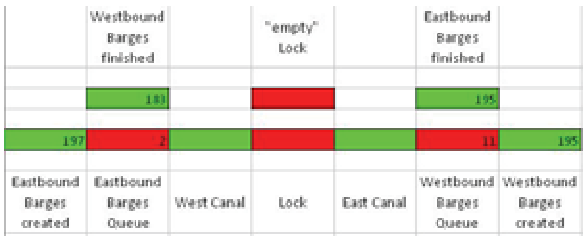


Figure 2. Visualisation

of the colours from green to red realise the current state of the lock and the channel. Further interesting values like simulation time, Queue contents or created barges are changing during the simulation run. For the experiments the visualisation can be disabled to increase the speed of the simulation.

B-Task: In this task the model is validated through five different sets of deterministic data. The changing of the data is easily possible through the small user interface. All five data sets lead to the correct results.

C-Task: In task c different variance reduction experiments should be done. Therefore the interarrival time of the barges is set to an exponential distribution with an expected value of 75 minutes. For this purpose uniform distributed values are created on a special spreadsheet before the simulation is starting. During the simulation these uniform distributed values are used one after another. The selection of the values is controlled by two global counters which are increased during the simulation. This approach is chosen because of the simple use and selection of random numbers for the experiments with antithetic variates and common random numbers. For each pair the same random numbers are used. With the VBA and Excel built-in function “Randomize” the seed of the random number generator is changed and values for a new independent run can be created. The inverse function of the exponential distribution is not implemented as built-in function in Excel and VBA. For this reason the inverse function is directly programmed and the uniform distributed values are used to calculated and create exponential distributed values.

Table 1 shows the results of the average barge transit time for three independent experiments on a pooled basis of 100 independent simulation runs. Column 3 and 4 show the length of the confidence interval and the standard deviaton of each run. The parameters “Eastmax” and “Westmax” are set to 5 in both cases.

Table 2 shows the results for the same experiment with antithetic random variates. The use of this varianve reduction method leads to a reduction of the confidence interval length which can be seen in column 5 of Table 1.

	Mean	CI	Std.	Reduction
Run 1	411,2	+/-24,0	103,0	41,2%
Run 2	415,9	+/-21,6	92,7	46,5%
Run 3	408,5	+/-18,6	80,1	60,4%

Table 1. Results of Antithetic Random Variates

	Mean	CI	Std.
Run 1	391,8	+/-40,8	175,4
Run 2	400,4	+/-40,3	173,3
Run 3	397,2	+/-47,0	202,2

Table 2. Results of 100 independent replications

	Mean	CI	Std.
Run 1	-40,4	+/-63,7	273,9
Run 2	39,3	+/-66,7	286,9
Run 3	50,0	+/-64,7	278,1

Table 3. Difference of 50 pairs

Table 3 shows the results for the difference of the mean for “Eastmax” and “Westmax” equal to 5 minus “Eastmax” and “Westmax” equal to 6. The difference is not significant and therefore the null hypothesis has to be rejected.

	Mean	CI	Std.	Reduction
Run 1	53,2	+/-7,5	32,2	88,3%
Run 2	61,9	+/-7,5	32,4	88,7%
Run 3	52,0	+/-7,4	31,9	88,5%

Table 4. Difference with Common Random Numbers

Table 4 shows the results for the same experiment with common random numbers. The variance reduction leads to a great reduction of the confidence interval length. The difference of the mean is significant for all three simulation runs and therefore the null hypothesis cannot be rejected.

Resumé: This work shows a new and different approach of solving comparisons. Visual Basic for Applications (VBA) is used to programm the class modules of source, queue, server and sink. Furthermore, the event logic and the event handler are directly programmed and no existing tool, software or simulator is used. Therefore the model is created without any graphical aid.

Corresponding author: Thomas Löscher,
Department of Analysis and Scientific Computing
Vienna University of Technology
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
thomas@loescher.at

Received: June 20, 2007
Reviewed: September 19, 2007
Accepted: October 10, 2007

Modeling ARGESIM Benchmark ‘C13 Crane and Embedded Control’ with Hybrid System Approach using Scicos

Masoud Najafi, Ramine Nikoukhah

Simulator: Scilab (`www.scilab.org`) is a free and open source simulation environment used for scientific computing. Scicos (`www.scicos.org`) is a toolbox of Scilab used for modeling and simulation of very general hybrid systems [1]. Scilab/Scicos can be considered as the counterparts of Matlab/Simulink. In Scicos which is a block diagram oriented simulator both discrete-time and continuous-time systems or in general hybrid systems can be modeled and simulated. Scicos has several toolboxes for predefined blocks and the user can define new blocks. Scicos has been extended to support a subset of Modelica language to define the behavior of blocks and the simulate component based models.

Model: The Modelica language provides the possibility of writing the nonlinear model directly as DAE. Thus we have used a generic Modelica block to build a block modeling the nonlinear car including the motor and the brake, see Fig. 1, 2. The inputs of the nonlinear model are *fc_des* (Desired force), *fd* (disturbance forces), and *BrakeStatus*. The outputs are *xc*, α and *xL*. The Modelica compiler of Scicos then compiles the program and generates a C code which can be integrated by the numerical solver. The Modelica program used used in the block follows.

```
class Nonlinear
...
equation
  DC=if (BrakeStatus>0.5) then dc
         else dc_Brake;
  der(fc)=-4*(fc-fc_des);
  Sa=Modelica.Math.sin(a);
  Ca=Modelica.Math.cos(a);
  der(xc)=w;
  der(w)*(mc+ml*Sa^2.0)=
         -DC*w+fc+fd*Sa^2.0+...;
  der(a)=da;
  r*r*der(da)*(ml*Sa^2+mc)=
         (fd*mc/ml-fc+DC*w)*...;
  xL=xc+r*Sa;
end Nonlinear;
```

In Scicos, the user is allowed to use standard Scicos blocks and Modelica blocks in the same diagram. Thus the rest of the model, *i.e.*, the controller, input signals, the display mechanism, the car supervisor which operates concurrently with the car controller have been implemented with standard Scicos blocks

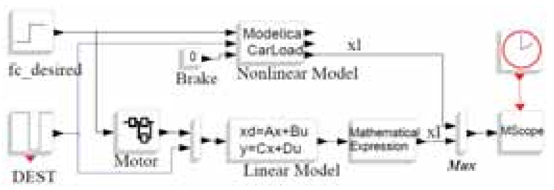


Figure 1. Scicos Model for A-task

such as event sources, discrete linear systems, counters, logic gates, comparators, see Fig. 3.

Scicos is inherently an event based simulator. At every discrete event time instant discrete parts of the model are updated. Between two discrete events only continuous parts (here the DAE and linear parts) are integrated. Each time there is discontinuity in input signals or in the continuous part and at sampling time instants, the Scicos simulator cold-restarts the numerical solver [2].

A-Task: The linear system including the discrete observer, the controller, and the motor has been implemented directly with Scicos standard blocks, whereas the nonlinear part, as aforesaid, is implemented with Modelica, see Fig. 1.

In order to compute the steady state values, a Scilab program calls the Scicos diagram of Fig.1 in batch mode with different values of disturbances. The steady state errors in x_L after 2000 seconds are given in table 1.

```
load Task1.cos;
dest=[-750,-800,-850];
for i=1:3
  Info=list();
  %scicos_context.DEST=dest(i);
  Info=scicos_simulate(scs_m,Info,
      %scicos_context,'nw');
  Y=Info(2).state.outtb(10);
  diff(i)=Y(1)-Y(2);
end
```

B-Task: The complete model of the car, controller, input signals, etc. are given in Fig.2. The supervisor including the brake condition checking is shown in Fig. 3. At each sampling time instant, if the desired force is less than the brake condition, the counter is incremented by one. Whenever the counter

SNE 17/3-4, December 2007

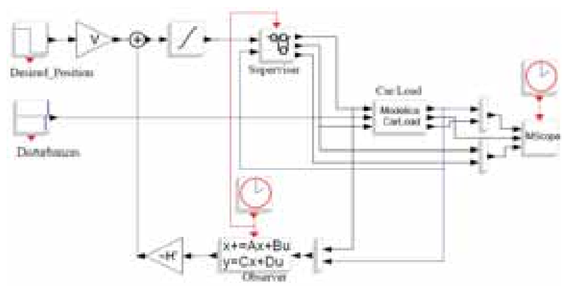


Figure 2. Scicos diagram used for B-task and C-task

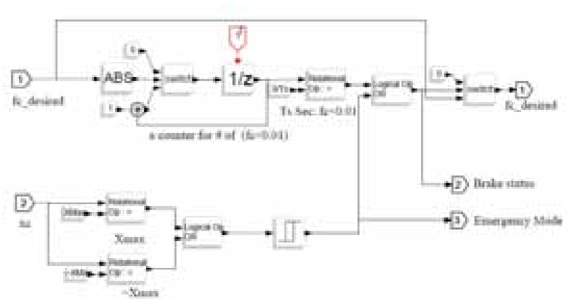


Figure 3. Supervisor block used in diagram of Fig. 2

reaches $3/Ts$, the brake status is activated, which in turn causes a change in the friction coefficient in the Modelica model of the car. Fig.4a shows the result for the car and load positions (x_C, x_L), Angle (α), and the brake status. The brake is active at $t = 35.5$ for about 0.5 seconds.

C-Task: The diagnostic mechanism has been implemented with two comparison blocks and a OR block, as shown in Fig. 3. Whenever x_C becomes less than $x_{\min}$ or greater than $x_{\max}$, the OR blocks becomes active and change the outputs of the hysteresis block. The output of the hysteresis block remains high, regardless of the input state. This mechanism can also be implemented easily within the Modelica program. Fig. 4b shows the results for the car and load positions (x_C, x_L), Angle (α), and the brake status and the emergency stop. The car does an emergency stop at t=47.94 Sec.

Although it is possible to implement all supervisor mechanism with Modelica, we decided to use standard Scicos blocks. The main reason lies in the fact that unfortunately a Modelica program cannot be synchronized with an external event sources. An implementing the counter in the Modelica program requires using an additional event source which doubles the number of the solver cold re-starts and reduces the simulation speed.

Dest	Δx_L (non-linear to linear)
-750	-0.000307
-800	0.004533
-850	0.000919

Table 1: Steady state errors in x_L after 2000 seconds

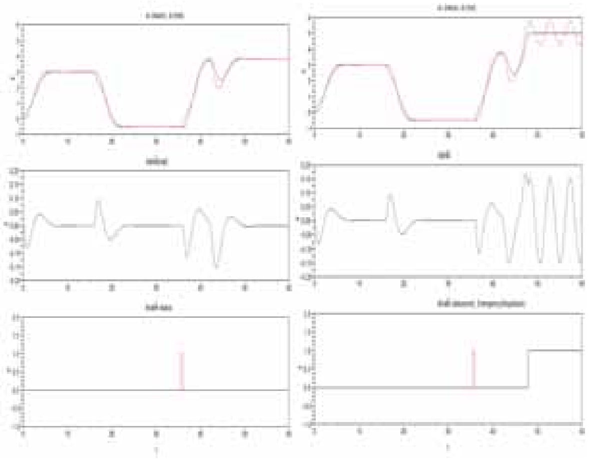


Figure 4. Simulation result for (a) B task, (b) C task

Resumé: The proposed solution uses different modeling methods available in Scicos. The linear continuous time blocks are modeled with blocks in the linear palette of Scicos, the discrete time observer is implemented using a periodic event source and a linear discrete system, and the nonlinear part has been implemented using the Modelica language. All these parts give a nonlinear hybrid DAE system of equation which can be simulated by the Scicos simulator.

References

[1] S. L. Campell, J. Chancelier, R. Nikoukhah. *Modeling and simulation in Scilab/Scicos*. Springer, 2005.

[2] M. Najafi. *The Numerical Solver for the Simulation of the Hybrid Dynamical Systems*. Doctor of science thesis, Paris XII University, 2005.

Corresponding author: Masoud Najafi
INRIA-Rocquencourt, Domaine de Voluceau,
BP 105, 78153, Le Chesnay, France

Received: September 10, 2007
Accepted: September 20, 2007

A Classic MATLAB-based Solution of ARGESIM Benchmark C18 'Identification of Non-linear Dynamic Relations'

Aleš Belič, University of Ljubljana, Slovenia; *ales.belic@fe.uni-lj.si*

Simulator: MATLAB 5.3 with Neural networks toolbox, running on Debian Linux 3.0 was used to solve three tasks: identification with linear model, identification with parallel structure of linear model and ANN, and identification with dynamical ANN.

Model: In identification of dynamical systems, linear methods are often applied first, because of well defined theoretical backgrounds of methods, even though it is clear that the system has non-linear characteristics. Therefore, we started with least-squares identification of discrete-time dynamical model. Non-linear identification requires large databases when we have to identify model parameters as well as the non-linearity type. Therefore, several model structures were developed, to reduce the necessary amount of data. One of such structures is parallel combination of linear dynamical model and non-linear statical model (see Figure 1). For non-linear model 2 layered ANN with 7 neurons, having tangens sigmoid function on the first layer and one neuron with linear function on the output was used.

Alternative to parallel structure would be serial structure, however, identification of serial structure is more complex, since both models must be identified in parallel, whereas for parallel structure dynamical model is identified first, and non-linear model is identified with goal to reduce the error of the linear model. Considering the quality of prediction achieved by parallel structure and available data, one must decide whether it is sensible to identify the system with dynamical non-linear structure, such as feed-forward ANN with feedback, or several other dynamical ANN structures, fuzzy models, splines, etc. In our case dynamical feed-forward 2 layered ANN was used with 10 neurons with tangens sigmoid functions on the first layer and one neuron with linear function on the output layer. The output of the network was delayed for one and two samples and fed back to the input layer of the network.

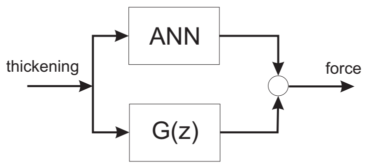


Figure 1. Parallel structure for identification of non-linear systems

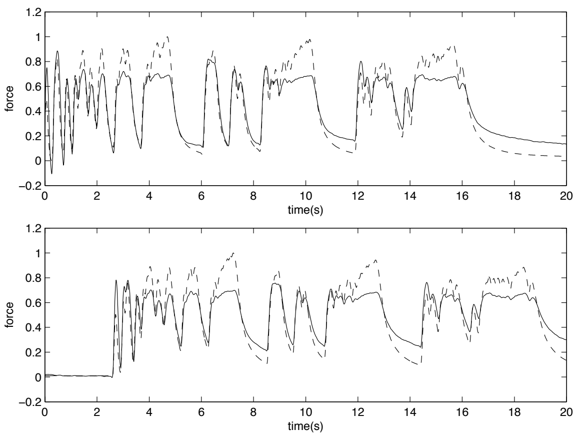


Figure 2. Simulated (solid line) in compare with measured force (dashed line). Training data set above, validation data set below.

A-Task: First identification with linear dynamical model was tried. Matlab *arx* function was used and 2nd order discrete-time model was identified (Eq. 1).

$$G(z) = \frac{0.5289z^2 - 0.5206z}{z^2 - 1.586z^1 + 0.5985} \tag{1}$$

Model simulation with respect to measured data is presented in Figure 2. As can be seen in Figure 2, linear second order model can describe the general system dynamics, however, the details are not matched. Calculated correlation coefficient of measured and simulated force is 0.95 for training data set and 0.94 for validation set. However, the trend of the error between simulated and measured force is -0.177s^{-1} for training set and -0.02 s^{-1}.

B-Task: The following code in MATLAB was used:

```
E = T-y';
net = ...
   newff(minmax(P),[7,1],{'tansig','purelin'});
net1 = train(net,P,E);
y1 = sim(net1,P);
plot(t,y+y1',t,T,'--')
```

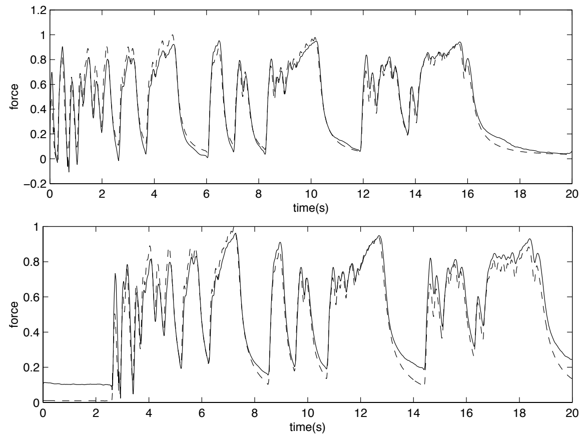


Figure 3. Simulation of the parallel structure of linear dynamical model and the ANN (solid line) in compare with measured force (broken line). Training data set above, validation data set below.

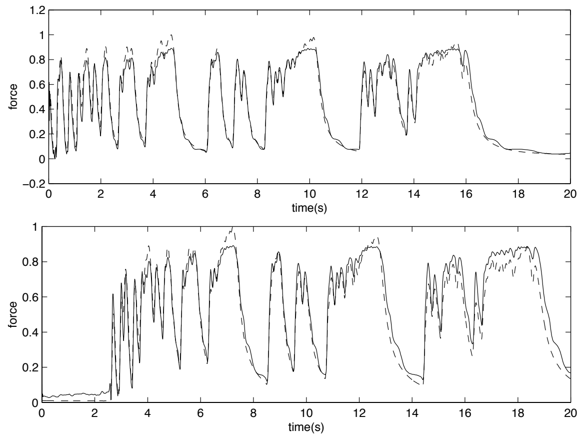


Figure 4. Simulation of the neural network model (solid line) in compare with measured force (broken line). Training data set above, validation data set below.

In the code, E represents the difference between real system's measurements `T` and linear model simulation `y`. Next, a network structure net is created with 7 neurons on the first layer and 1 neuron on the output layer, and is trained according to the system input `P` and target `T`. The network and the linear dynamical model are then simulated in parallel and the result of the hybrid system is shown in Figure 3.

Calculated correlation coefficient increases to 0.98 for training and validation set. However, the trend in error remains -0.151s^{-1} for training set and -0.07 s^{-1} for validation set.

C-Task: The following code was used for the ANN model training:

```
net = ...
   newff([0 1],[10,1],{'tansig','purelin'});
net.layerconnect = [0 1;1 0];
net.layerweights{1,2}.delays = [1,2];
net.inputweights{1,1}.delays = [1,2];
net.trainparam.epochs = 50;
net.trainparam.show = 1;
net2 = train(net,con2seq(P),con2seq(T));
y2 = seq2con(sim(net2,con2seq(P)));
```

The results are shown in figure 4.

Correlation coefficient for model M3 between measured and simulated data for training set is slightly raised (0.99) while it remains the same for validation set (0.98). The problem still remains the trend in error between simulated and measured data. For training set, the trend is -0.155 s^{-1}, and for validation set -0.178 s^{-1}.

Resumé: It can be concluded that relation between muscle belly displacement and muscle force has a non-linear characteristic, since it is not possible to model the details with model M1 while the model M1 is relatively successful in describing the system dynamics. With both non-linear models M2 and M3 it is possible to achieve much better prediction of details as well as system dynamics. However, the problem with trend in error between measured data and simulation remains, regardless of the model used for simulation (M1, M2, or M3). The values of trends in error are relatively independent on the model used for simulation. Thus it can be concluded that not all the information of muscle force is present in the muscle belly thickening signal. In spite of missing information, a high correlation of simulated and measured force can be obtained. Although muscle belly thickening signal can provide relatively good information on muscle force dynamics for frequencies up to 5Hz, it cannot serve as an alternative measurement for absolute muscle force measurements.

Corresponding author: Aleš Belič,
Univ. of Ljubljana, Fac. of Electrical Engineering,
Tržaška 25, 1000 Ljubljana, Slovenia
ales.belic@fe.uni-lj.si

Received: September 14, 2007
Revised: October 10, 2007
Accepted: October 15, 2007

Solution to ARGESIM Benchmark C18 'Identification of Nonlinear Dynamical Relations' using Excel with a Neural Networks Add-In

Sašo Blažič, University of Ljubljana, Slovenia; *saso.blazic@fe.uni-lj.si*

Simulator: It is known for quite some time that artificial neural networks (ANNs) are capable of modelling very complex nonlinear systems. Hence, they have become very popular and widely used in a variety of applications. A reasonable conclusion was that ANNs should be somehow accessible to a wider public. Many companies started developing Add-Ins for Microsoft Excel that has become the industry-standard data analysis and modelling tool. These Add-Ins are usually very easy to use and a user without much experience with ANNs is capable of training a network and using it for prediction or classification purposes. The drawback is that the user is very limited in customising the ANN.

Two tasks (identification with parallel structure of a linear model and an ANN, and identification with a dynamical ANN) were solved in the paper with the following software configuration: Microsoft Office Excel 2003, and NeuralTools, Neural Net Add-In for Microsoft Excel, Version 1.1.0 – Professional Edition, Palisade Corporation.

Model: When using NeuralTools, neural networks are developed and used in four steps:

- *Data preparation*. The data used in NeuralTools are defined in data sets. A Data Set Manager is used to set up data sets so they can be used over and over again with neural networks.
- *Training*. With training, a neural network is generated from a data set comprised of cases (input vectors) with known output values. This data often consists of historical cases for which the values of output/dependent variable are known.
- *Testing*. With testing, a trained neural network is tested to see how well it does at predicting known output values. The data used for testing is usually a subset of historical data. This subset was not used in training the network. After testing, the performance of the network is measured by statistics such as the percentage of the known answers it correctly predicted.
- *Prediction*. A trained neural network is used to predict unknown output values. Once trained and tested, the network can be used as needed to predict outputs for new case data.

Each of the steps is done by simply clicking the appropriate icon and setting a few parameters.

NeuralTools supports different neural network configurations to give the best possible predictions. For classification/category prediction (where the dependent variable is a category type), two types of networks are available: Probabilistic Neural Networks (PNN) and Multi-Layer Feedforward Networks (MLFN). Numeric prediction can be performed using MLFN networks, as well as Generalized Regression Neural Networks (GRNN), which are closely related to PNN networks.

NeuralTools makes selecting a network configuration easy by offering a Best Net search. When selected, NeuralTools will train and test a variety of neural network configurations to generate the one that gives the best predictions for the data. The best configuration is determined based on testing data.

A The linear model used is the same as the one proposed in the solution by Aleš Belič, identified in MATLAB, and not separately investigeated in EXCEL:

$$G(z)=\frac{y_{lin}}{u}=\frac{0.5289z^2-0.5206z}{z^2-1.586z^1+0.5985} \qquad (1)$$

There y_{lin} is the output of the linear part, with input variable u (thickening) and output variable y (force).

B-Task: In this task the parallel connection of an LTI system and a static ANN was used. The linear model is given in (1). The output of the model is $y_{lin}+y_{ANN}$, and the ANN is therefore trained with u at its output and the residual error $y-y_{lin}$ at its output.

Two columns are prepared in Excel. In the first column the input samples (from the identification signal) are prepared and in the second column comprises of the samples of $y-y_{lin}$. These two signals are previously prepared in MATLAB and imported into Excel. Then (identification) data set is defined with the Data Set Manager. The training is started with the Generalized Regression Neural Networks that does not require any designer parameters (whereas Multi-Layer Feedforward Networks require the number of nodes to be defined). (Actually, this is a very simple map-

SNE 17/3-4, December 2007

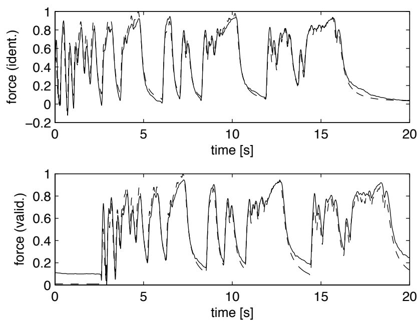


Figure 1. Simulation of the parallel structure of linear dynamical model and the ANN (solid line) in comparison with the measured force (dashed line). Training data set (top), validation data set (bottom).

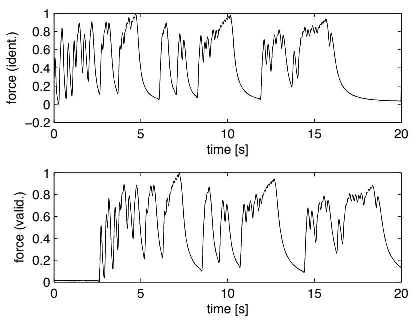


Figure 2. One-step-ahead prediction of the neural network model (solid line) in comparison with the measured force (broken line). Training data set – above, validation data set – below.

ping with one input and one output, and therefore any setting results in a similar network response.)

The ANN and the linear dynamical model are then simulated in parallel and the result of the hybrid system is shown in Figure 1. Calculated correlation coefficients for this case are 0.9805 and 0.9814 for the training and the validation data set, respectively.

C-Task: In this task the dynamical ANN is used. There are four inputs to the system: $y(k\text{-}1)$, $y(k\text{-}2)$, $u(k\text{-}1)$, and $u(k\text{-}2)$, and the output is $y(k)$. The ANN is trained on the identification data. The network response is simulated in the validation phase (with the new data).

NeuralTools does not enable training of dynamical neural networks directly. In the training phase only a static data set is permitted. In the validation phase we want to simulate the response of the ANN to a particular input signal which is again not possible directly with the NeuralTools. The software only offers the possibility of predicting the output to a static data set. In order to simulate a dynamical network one needs to feed back the delayed network outputs. But NeuralTools has a special feature that can be made good use of in order to simulate the dynamical network. It is possible to start the prediction in the “Live prediction” mode. This means that the output of the network is recalculated as soon as the input data change (in Excel worksheet). A simple Excel macro was written that iteratively copies network predictions and pastes them as inputs for the future outputs. Thus, outputs are effectively fed back and the network becomes dynamical.

NeuralTools with the addition of a simple macro are therefore capable of simulating the dynamical ANN. The problem is that in the training phase it is virtually impossible to achieve a stable dynamical ANN for this case. A vast amount of tests were made and the results were always the same: very good prediction ability (for identification and validation data) and locally unstable models when using feedback. It is well-known that the dynamical neural networks are much harder to train than the static ones. Although they can be trained using the same gradient-based algorithms that are used for static networks, the performance of the algorithms on dynamic networks can be much worse. If the simulation of a dynamical ANN is possible with some extensions, the training procedures turned out not to be suitable for training of the dynamical ANN for this case.

Since unusable simulation data were obtained, one-step-ahead predictions are shown in Figure 2.

Resumé: NeuralTools turned out to be quite effective in training static networks. The software is intuitive, very simple to use and suitable also for users without much experience and knowledge about artificial neural networks. This simplicity is also a certain drawback since an experienced user is very limited in customising the network. It turned out that the training algorithm of the software is not very suitable for dynamical networks and similar problems as the ones faced in our experiments can be encountered. It is true that the data used in this case are difficult for the identification since a slight change of the dynamics with time can be observed and also the input signal excitation is not very good at high frequencies.

Corresponding author: Sašo Blažič,
Univ. of Ljubljana, Faculty of Electrical Engineering,
Tržaška 25, 1000 Ljubljana, Slovenia
saso.blazic@fe.uni-lj.si

Received: September 14, 2007
Revised October 10, 2007
Accepted October 15, 2007

Book Reviews – Journal Reviews

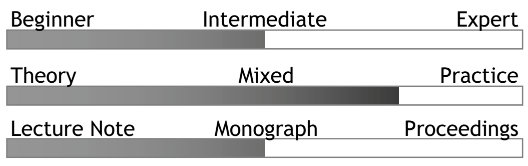
Book Reviews

Approximation of Large-Scale Dynamical Systems
Anthanasios C. Anotoulas
Society for Industrial and Applied Mathematics, 2005
479 + xv pages, ISBN 0-898-71529-6

Mathematical models are used to simulate, and sometimes control, the behaviour of physical and artificial processes such as the weather and very large-scale integration (VLSI) circuits. The increasing need for accuracy has led to the development of highly complex models. However, in the presence of limited computational, accuracy, and storage capabilities, model reduction (system approximation) is often necessary.

Approximation of Large-Scale Dynamical Systems provides a comprehensive picture of model reduction, combining system theory with numerical linear algebra and computational considerations. It addresses the issue of model reduction and the result-ing trade-offs between accuracy and complexity. Special attention is given to numerical aspects, simulation questions, and practical applications. This book is for anyone interested in model reduction. Graduate students and researchers in the fields of system and control theory, numerical analysis, and the theory of partial differential equations/computational fluid dynamics will find it an excellent reference.

The first two chapters are an introduction and present preliminaries giving sufficient background on the topic. In the book singular value decomposition methods (SVD) and Krylov subspace projection methods are clearly differentiated and are explained in detail in part III and IV. The book does require some familiarity with numerical linear algebra and may be difficult for a new comer in the area to follow. This book shows the concepts of reachability and observability and how they are applied to get a better understanding of dynamical systems.



Daniel Leitner, Vienna Univ. of Technology
dleitner@osiris.tuwien.ac.at

Differential Equations
Linear, Nonlinear, Ordinary, Partial
A. C. King, J. Billingham, S. R. Otto
Cambridge University Press, 2003
541 + xii pages, ISBN 0-521-01687-8

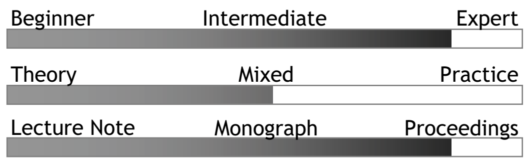
This book gives an introduction into the theory of differential equations and discusses various applications. It is divided into two parts: the first part deals with linear differential equations and the second, longer part with nonlinear ones and applications. In the first part solutions of differential equations are derived analytically. The methods used include power series, Green's functions, Sturm-Liouville theory, Fourier transforms and Laplace transforms. Also the properties of special functions, i.e. Legendre functions and Bessel functions, as solutions of the respective differential equations, are discussed.

The second part starts more theoretically with the existence and uniqueness properties of ordinary differential equations. Then three approaches suitable for nonlinear equations are presented: phase plane methods, group theoretical methods and asymptotic methods. At the end of the book properties of chaotic systems are discussed. Especially interesting for people working in modelling and simulation is the chapter on time-optimal control.

The appendixes give an overview of the mathematics necessary to understand the book. A very good feature is that the text gives MATLAB examples which can be used gain a better understanding. At the end of each chapter exercises are given, unfortunately without the (final) solution.

The different topics are presented mathematically rigorous, although some extensive proofs are only referenced. The chapters with the advanced topics vary greatly in their length and difficulty.

The topic selection makes this book is suitable for theoretical physics or applied mathematics students.



Peter Kristöffel, *krist@physik.htu.at*

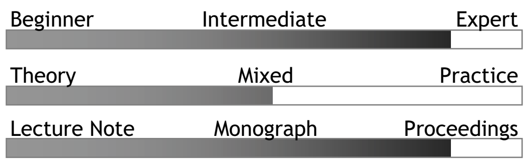
A Guided tour of Mathematical Methods for the Physical Sciences – 2nd edition
Roel Snieder
Cambridge University Press, 2004
507 + xiv pages, ISBN 0-521-83492-9

This book gives an introduction into almost all the mathematics a physicist ever needs. The mathematical tools are developed starting from examples which is a very motivating way of introducing undergraduate physics students to mathematics. Throughout the book many exercises are given in the text-flow.

Many different topics are covered in the 26 chapters of the book, although there is no red line in the sequence of the fields of study. Advanced readers may use single chapters as a nice introduction to fields in which they have no expertise yet. The main topics presented are: power series, coordinate systems, vector analysis, scale analysis, linear algebra, the Dirac delta function, Fourier analysis, complex analysis, Green's functions, tensors and perturbation theory. The second edition of the book contains three more chapters than the first edition. These additional chapters are about dimensional analysis, the asymptotic evaluation of integrals and variational calculus.

Throughout the whole book the mathematical objects are discussed on the basis of examples from many different fields of physics. Various properties of these objects are developed from problems arising in the physical sciences. It's refreshing to see that one can take so many different viewpoints to develop and apply mathematical theory. The book strongly is influenced by methods of applied mathematics and enriched by references to numerical mathematics. That no hints or solutions to the exercises are given is a minor fault. Many pictures help to get a better understanding of the fields presented.

The book is suited to accompany physics undergraduate students throughout their whole study and may be useful for teachers who look for problems and examples from physics.



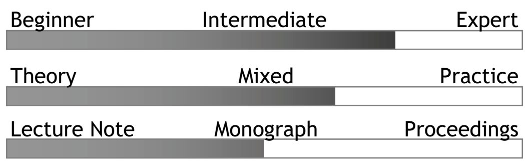
Peter Kristöffel, Vienna Univ. of Technology
krist@physik.htu.at

Cellular Neural Networks: Dynamics and Modelling
Angela Slavova
Mathematical Modelling: Theory and Applications, vol. 16
Kluwer Academic Publishers, 2003
220 + x pages, ISBN 1-402-01192-X

This book deals with new theoretical results for studying Cellular Neural Networks (CNNs), locally coupled neural networks introduced in 1988 by L. O. Chua and L. Yang, concerning its dynamical behaviour. New aspects of CNNs' applications are developed for modelling of some famous nonlinear partial differential equations arising in biology, genetics, neurophysiology, physics, ecology, etc. The analysis of CNNs' models is based on the harmonic balance method well known in control theory and in the study of electronic oscillators. Such phenomena as hysteresis, bifurcation and chaos are studied for CNNs.

The topics investigated in the book involve several scientific disciplines, such as dynamical systems, applied mathematics, mathematical modelling, information processing, biology and neurophysiology. The reader will find comprehensive discussion on the subject as well as rigorous mathematical analyses of networks of neurons from the view point of dynamical systems. The book contains interesting theoretical results on dynamics of CNNs along with examples illustrating the usefulness of CNNs for mathematical modelling in natural sciences.

The text is written as a textbook for senior undergraduate and graduate students in applied mathematics. Providing a summary of recent results on dynamics and modelling of CNNs, the book will also be of interest to all researchers in the area. It is divided into 2 parts, whereas the first one presents the basic theory of CNNs and some new results of the author concerning the dynamics of nonlinear CNNs. The second part of the book deals with approximation techniques, their usage and their degree of reliability. Furthermore several CNN models of equations coming from above mentioned fields are presented.



Christopher Mayer, Vienna Univ. of Technology
ccmayer@osiris.tuwien.ac.at

SNE News Section

Data & Quick Info

EUROSIM 2010
organised by CSSS
September 2010, Prague, Czech Republic

Contents

Info EUROSIM 2
Info EUROSIM Societies .... 3 - 6
Info ASIM 3
Info CROSSIM, CSSS, DBSS, FRANCOSIM 4
Info HSS, ISCS, PSCS, SIMS 5
Info SLOSIM, UKSIM, LSS CAE-SMSG, ROMSIM 6
Simulationist Lukas Schuler 7

Simulation News Europe is the official journal of EUROSIM and sent to most members of the EUROSIM Societies as part of the membership benefits. Furthermore **SNE** is distributed to other societies and to individuals active in the area of modelling and simulation. SNE is registered with ISSN 1015-8685. Circulation of printed version is 3000 copies.

SNE at Web recent issues of SNE are also available via internet at `www.argesim.org`. Members of EUROSIM Societies may have access to the SNE Archive. This special *News Section* compiles date from EUROSIM and EUROSIM societies: addresses, weblinks, officers of societies with function and email. This *EUROSIM Data & Quick Info* is published regularly in SNE issues (except for special issues).

SNE Reports Editorial Board

INFO: WWW.ARGESIM.ORG; sne@argesim.org

If you have any information, announcement, etc. you want to see published, please contact a member of the editorial board in your country or sne@argesim.org.

Editorial Information/Impressum - see front cover

Information EUROSIM

EUROSIM Federation of European Simulation Societies

General Information. *EUROSIM*, the Federation of European Simulation Societies, was set up in 1989. The purpose of EUROSIM is to provide a European forum for regional and national simulation societies to promote the advancement of modelling and simulation in industry, research, and development.

→ www.eurosim.info

Member Societies. EUROSIM members may be national simulation societies and regional or international societies and groups dealing with modelling and simulation. At present EUROSIM has eleven full members and three observer members:

ASIM	Arbeitsgemeinschaft Simulation *Austria, Germany, Switzerland*
CROSSIM	Croatian Society for Simulation Modeling *Croatia*
CSSS	Czech and Slovak Simulation Society *Czech Republic, Slovak Republic*
DBSS	Dutch Benelux Simulation Society *Belgium, Netherlands*
FRANCOSIM	Société Francophone de Simulation *Belgium, France*
HSS	Hungarian Simulation Society *Hungary*
ISCS	Italian Society for Computer Simulation *Italy*
PSCS	Polish Society for Computer Simulation *Poland*
SIMS	Simulation Society of Scandinavia *Denmark, Finland, Norway, Sweden*
SLOSIM	Slovenian Simulation Society *Slovenia*
UKSIM	United Kingdom Simulation Society *UK, Ireland*
CEA-SMSG	Spanish Modelling and Simulation Group *Spain, Observer Member*
LSS	Latvian Simulation Society *Latvia, Observer Member*
ROMSIM	Romanian Society for Modelling and Simulation, *Romania, Observer Member*

Contact addresses, weblinks and officers of the societies may be found in the information part of the societies.

EUROSIM board/EUROSIM officers. EUROSIM is governed by a board consisting of one representative of each member society, president and past president, and representatives for SNE and SIMPRA. The President is nominated by the society organising the next EUROSIM Congress. Secretary and Treasurer are elected out of members of the Board.

President	Mikuláš Alexík (CSSS), *alexik@frtk.fri.utc.sk*
Past president	Borut Zupančič (SLOSIM) *borut.zupancic@fe.uni-lj.si*
Secretary	Peter Fritzson (SIMS) *petfr@ida.liu.se*
Treasurer	Felix Breitenecker (ASIM) *felix.breitenecker@tuwien.ac.at*
SIMPRA Repres.	Jürgen Halin *halin@iet.mavt.ethz.ch*
SNE Repres.	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*

SNE – Simulation News Europe. EUROSIM societies are offered to distribute to their members the journal *Simulation News Europe* (SNE) as official membership journal. SNE is a scientific journal with reviewed contributions in the *Notes Section* as well as a membership newsletter for EUROSIM with information from the societies in the *News Section*. Publisher are EUROSIM, ARGESIM and ASIM.

Editor-in-chief	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*

→ www.argesim.org, menu SNE

→ www.asim-gi.org, menu International

EuroSim Congress. EUROSIM is running the triennial conference series EUROSIM Congress. The congress is organised by one of the EUROSIM societies. EUROSIM 2010 will be organised by CSSS in Prague, September 5-10, 2010.

Information	Mikulas Alexik (CSSS) *alexik@frtk.utc.sk*
Chair OC 2010	Miroslav Šnorek *snorek@fel.cvut.cz*

→ www.eurosim.org

ASIM German Simulation Society Arbeitsgemeinschaft Simulation

ASIM (Arbeitsgemeinschaft Simulation) is the association for simulation in the German speaking area, servicing mainly Germany, Switzerland and Austria. ASIM was founded in 1981 and has now about 700 individual members, and 30 institutional or industrial members. Furthermore, ASIM counts about 300 affiliated members.

→ www.asim-gi.org with members' area
info@asim-gi.org, admin@asim-gi.org
ASIM – Inst. f. Analysis and Scientific Computing
Vienna University of Technology
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria

ASIM Working Groups. ASIM, part of GI - Gesellschaft für Informatik, is organised in Working Groups, dealing with applications and comprehensive subjects:

GMMS	Methods in Modelling and Simulation Peter Schwarz, *schwarz@eas.iis.fhg.de*
SUG	Simulation in Environmental Systems Wittmann, *wittmann@informatik.uni-hamburg.de*
STS	Simulation of Technical Systems H.T.Mammen, *Heinz-Theo.Mammen@hella.com*
SPL	Simulation in Production and Logistics Sigrid Wenzel, *s.wenzel@uni-kassel.de*
SVS	Simulation of Transport Systems U. Brannolte, *Brannolte@bauing.uni-weimar.de*
SBW	Simulation in OR C. Böhnlein, *boehnlein@wiinf.uni-wuerzburg.de*
EDU	Simulation in Education/Education in Simulation W. Wiechert, *wiechert@simtec.mb.uni-siegen.de*

ASim Publications

SNE – Simulation News Europe. ASIM is publishing (co-publishing) SNE, which is regularly published and sent to all ASIM members (as part of their membership; 900 issues) and for promotion purposes (300 issues). Since 2006, the ASIM Working Groups publish *SNE Special Issues* with state-on-the-art reports on modelling and simulation in their workscope.

ASIM News. In December 2005, the ASIM Nachrichten has been replaced by an electronic news-letter - ASIM Newsletter. Editors are Th. Pawletta and C. Deatcu, Univ. Wismar, pawel@mb.hs-wismar.de.

ASIM Notes/ASIM Mitteilungen. The trademark ASIM Mitteilungen (ASIM Note) stands for all publications of ASIM and of the the ASIM Working Groups. Each publication gets an identification as ASIM Notes, independent of the publisher, and independent of the publication medium (printed books, CD, Web). ASIM Notes range from printed books (with CDs) published by Springer, via workshop publication published in SNE or ARGESIM, to compiled abstracts publishes at the ASIM weberver.

ASIM Books. ASIM co-operates with the SCS Publishing House e.V., with ARGESIM (Vienna University of Technology), and with Shaker Verlag Aachen in publication of two book series (Fortschritte in der Simulationstechnik - Frontiers in Simulation and Fortschrittsberichte Simulation - Advances in Simulation) and in publication of Proceedings. Publications in these series range from monographs via proceedings to PhD theses.

ASIM Board and Officers: The ASIM board consists of officers (elected all three years), of the chairpersons of the ASIM Working Groups (independently elected all three years), and of co-opted specialists.

Prsident	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*
Vice presidents	Sigrid Wenzel *s.wenzel@uni-kassel.de*
	Thorsten Peawletta, *pawel@mb.hs-wismar.de*
Secretary	Claus-Burkhard Böhnlein, *boehnlein@wiinf.uni-wuerzburg.de*
Treasurer	Ingrid Bausch-Gall, *Ingrid@Bausch-Gall.de*
Membership affairs	S. Wenzel, *s.wenzel@uni-kassel.de*
	W. Maurer, *werner.maurer@zhwin.ch*
	I. Bausch-Gall, *Ingrid@Bausch-Gall.de*
	F. Breitenecker (*mail address above*)
Universities	W. Wiechert *wiechert@simtec.mb.uni-siegen.de*
Industry	S. Wenzel, *s.wenzel@uni-kassel.de*
	K. Panreck, *Klaus.Panreck@hella.com*
Conferences	Klaus Panreck *Klaus.Panreck@hella.com*
	Albrecht Gnauck *albrecht.gnauck@tu-cottbus.de*
Publications	Th. Pawletta, *pawel@mb.hs-wismar.de*
	F. Breitenecker (*mail address above*)
Repr. EUROSIM	F. Breitenecker (*mail address above*)
Deputy	W. Wiechert *wiechert@simtec.mb.uni-siegen.de*
Edit. Board SNE	Thorsten Pawletta, *pawel@mb.hs-wismar.de*
Web EUROSIM	Anna Mathe, *anna.mathe@tuwien.ac.at*

CroSSim – Croatian Society for Simulation Modelling

CroSSim-*Croatian Society for Simulation Modelling* was founded in 1992 as a non-profit society with the goal to promote knowledge and use of simulation methods and techniques and development of education. CroSSim is a full member of EuroSim since 1997.

→ www.eurosim.info

✉ CroSSim / Jadranka Božikov
Andrija Stampar School of Public Health,
Medical School, University of Zagreb
Rockefeller St. 4, HR-10000 Zagreb, Croatia

President	Jadranka Božikov, *jbozikov@snz.hr*
Vice president	Vesna Dušak, *vdusak@foi.hr*
Secretary	Vesna Bosilj-Vukšić, *vbosilj@efzg.hr*
Executive board members	Vlatko Čerić, *vceric@efzg.hr*
	Tarzan Legović, *legovic@irb.hr*
Repr. EuroSim	Jadranka Bozikov, *jbozikov@snz.hr*
Edit. Board SNE	Jadranka Bozikov, *jbozikov@snz.hr*
Web EuroSim	Jadranka Bozikov, *jbozikov@snz.hr*

CSSS – Czech and Slovak Simulation Society

CSSS -The *Czech and Slovak Simulation Society* has about 150 members working in Czech and Slovak national scientific and technical societies (*Czech Society for Applied Cybernetics and Informatics*, *Slovak Society for Applied Cybernetics and Informatics*). The main objectives of the society are: development of education and training in the field of modelling and simulation, organising professional workshops and conferences, disseminating information about modelling and simulation activities in Europe. Since 1992, CSSS is full member of EuroSim.

→ www.fit.vutbr.cz/CSSS

✉ CSSS / Miroslav Šnorek, CTU Prague
FEE, Dept. Computer Science and Engineering,
Karlovo nam. 13, 121 35 Praha 2, Czech Republic

President	Miroslav Šnorek, *snorek@fel.cvut.cz*
Vice president	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Treasurer	Evžen Kindler, *ekindler@centrum.cz*
Scientific Secr.	A. Kavička, *Antonin.Kavicka@upce.cz*
Repr. EuroSim	Miroslav Šnorek, *snorek@fel.cvut.cz*
Deputy	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Edit. Board SNE	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Web EuroSim	Petr Peringer, *peringer@fit.vutbr.cz*

DBSS – Dutch Benelux Simulation Society

The Dutch Benelux Simulation Society (DBSS) was founded in July 1986 in order to create an organisation of simulation professionals within the Dutch language area. DBSS has actively promoted creation of similar organisations in other language areas. DBSS is a member of EuroSim and works in close cooperation with its members and is further affiliated with SCS International, IMACS, and the Chinese Association for System Simulation and the Japanese Society for Simulation Technology.

→ www.eurosim.info

✉ DBSS / A. W. Heemink
Delft University of Technology, ITS - twi,
Mekelweg 4, 2628 CD Delft, The Netherlands

President	A. Heemink, *a.w.heemink@its.tudelft.nl*
Vice president	W. Smit, *smitnet@wxs.nl*
Treasurer	W. Smit, *smitnet@wxs.nl*
Secretary	W. Smit, *smitnet@wxs.nl*
Repr. EuroSim	A. Heemink, *a.w.heemink@its.tudelft.nl*
Deputy	W. Smit, *smitnet@wxs.nl*
Edit. Board SNE	A. Heemink, *a.w.heemink@its.tudelft.nl*

FrancoSim – Société Francophone de Simulation

FrancoSim was founded in 1991 and aims to the promotion of simulation and research, in industry and academic fields. Francosim operates two poles.

- Pole Modelling and simulation of discrete event systems. Pole Contact: *Henri Pierreval, pierreva@imfa.fr*
- Pole Modelling and simulation of continuous systems. Pole Contact: *Yskandar Hamam, y.hamam@esiee.fr*

→ www.eurosim.info

✉ FRANCOSIM / Yskandar Hamam
Groupe ESIEE, Cité Descartes,
BP 99, 2 Bd. Blaise Pascal,
93162 Noisy le Grand CEDEX, FRANCE

President	Yskandar Hamam, *y.hamam@esiee.fr*
Treasurer	François Rocaries, *f.rocaries@esiee.fr*
Repr. EuroSim	Yskandar Hamam, *y.hamam@esiee.fr*
Edit. Board SNE	Yskandar Hamam, *y.hamam@esiee.fr*

HSS – Hungarian Simulation Society

The Hungarian Member Society of EUROSIM was established in 1981 as an association promoting the exchange of information within the community of people involved in research, development, application and education of simulation in Hungary and also contributing to the enhancement of exchanging information between the Hungarian simulation community and the simulation communities abroad. HSS deals with the organization of lectures, exhibitions, demonstrations, and conferences.

→ www.eurosim.info

✉ HSS / András Jávor,
Budapest Univ. of Technology and Economics,
Sztoczek u. 4, 1111 Budapest, HUNGARY

President	András Jávor, *javor@eik.bme.hu*
Vice president	Gábor Szűcs, *szucs@itm.bme.hu*
Secretary	Ágnes Vigh, *vigh@itm.bme.hu*
Repr. EUROSIM	András Jávor, *javor@eik.bme.hu*
Deputy	Gábor Szűcs, *szucs@itm.bme.hu*
Edit. Board SNE	András Jávor, *javor@eik.bme.hu*
Web EUROSIM	Gábor Szűcs, *szucs@itm.bme.hu*

ISCS – Italian Society for Computer Simulation

The Italian Society for Computer Simulation (ISCS) is a scientific non-profit association of members from industry, university, education and several public and research institutions with common interest in all fields of computer simulation.

→ www.eurosim.info

✉ ISCS / Mario Savastano,
c/o CNR - IRSIP,
Via Claudio 21, 80125 Napoli, ITALY

President	MarioSavastano, *mario.savastano@unina.it*
Vice president	F. Maceri, *Franco.Maceri@uniroma2.it*
Secretary	Paola Provenzano, *paola.provenzano@uniroma2.it*
Treasurer	Pasquale Arpaia
Repr. EUROSIM	F. Maceri, *Franco.Maceri@uniroma2.it*
Edit. Board SNE	Mario Savastano, *mario.savastano@unina.it*

PSCS – Polish Society for Computer Simulation

PSCS was founded in 1993 in Warsaw. PSCS is a scientific, non-profit association of members from universities, research institutes and industry in Poland with common interests in variety of methods of computer simulations and its applications. At present PSCS counts 264 members.

→ www.ptsk.man.bialystok.pl

✉ PSCS / Leon Bobrowski, c/o IBIB PAN,
ul. Trojdena 4 (p.416), 02-109 Warszawa, Poland

President	Leon Bobrowski, *leon@ibib.waw.pl*
Vice president	A. Chudzikiewicz, *ach@it.pw.edu.pl*
Treasurer	Z. Sosnowski, *zenon@ii.pb.bialystok.pl*
Secretary	Zdzislaw Galkowski, *Zdzislaw.Galkowski@simr.pw.edu.pl*
Repr. EUROSIM	Leon Bobrowski, *leon@ibib.waw.pl*
Deputy	A.Chudzikiewicz, *ach@it.pw.edu.pl*
Edit. Board SNE	Z.Sosnowski, *zenon@ii.pb.bialystok.pl*

SIMS – Scandinavian Simulation Society

SIMS is the *Scandinavian Simulation Society* with members from the four Nordic countries Denmark, Finland, Norway and Sweden. The SIMS history goes back to 1959. SIMS practical matters are taken care of by the SIMS board consisting of two representatives from each Nordic country. Iceland will be represented by one board member.

SIMS Structure. SIMS is organised as federation of regional societies. There are FinSim (Finnish Simulation Forum), DKSIM (Dansk Simuleringsforening) and NFA (Norsk Forening for Automatisering).

→ www.scansims.org

✉ SIMS/Peter Fritzson, IDA, Linköping University,
58183, Linköping, Sweden

President	Peter Fritzson, *petfr@ida.liu.se*
Treasurer	Vadim Engelson, *vaden@ida.liu.se*
Repr. EUROSIM	Peter Fritzson, *petfr@ida.liu.se*
Edit. Board SNE	Esko Juuso, *esko.juuso@oulu.fi*
Web EUROSIM	Vadim Engelson, *vaden@ida.liu.se*

SLOSIM – Slovenian Society for Simulation and Modelling

SLOSIM - Slovenian Society for Simulation and Modelling was established in 1994 and became the full member of EUROSIM in 1996. Currently it has 69 members from both slovenian universities, institutes, and industry. It promotes modelling and simulation approaches to problem solving in industrial as well as in academic environments by establishing communication and cooperation among corresponding teams.

→ `msc.fe.uni-lj.si/SLOSIM`

⌧ SLOSIM / Rihard Karba, Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, Slovenia

President	Rihard Karba, *rihard.karba@fe.uni-lj.si*
Vice president	Leon Žlajpah, *leon.zlajpah@ijs.si*
Secretary	Aleš Belič, *ales.belic@fe.uni-lj.si*
Treasurer	Milan Simčič, *milan.simcic@fe.uni-lj.si*
Repr. EUROSIM	Rihard Karba, *rihard.karba@fe.uni-lj.si*
Deputy	Borut Zupančič, *borut.zupancic@fe.uni-lj.si*
Edit. Board SNE	Rihard Karba, *rihard.karba@fe.uni-lj.si*
Web EUROSIM	Aleš Belič, *ales.belic@fe.uni-lj.si*

UKSIM – United Kingdom Simulation Society

UKSIM has more than 100 members throughout the UK from universities and industry. It is active in all areas of simulation and it holds a biennial conference as well as regular meetings and workshops.

→ `www.uksim.org.uk`

⌧ UKSIM / Alessandra Orsoni,
Kingston Business School, Kingston Hill,
Kingston-Upon-Thames, Surrey, KT2 7LB, UK

President	David Al-Dabass, *david.al-dabass@ntu.ac.uk*
Secretary	Alessandra Orsoni, *A.Orsoni@kingston.ac.uk*
Treasurer	B. Thompson, *barry@bjtcon.ndo.co.uk*
Membership chair	K. Al-Begain, *kbegain@glam.ac.uk*
Univ. liaison chair	R. Cheng, *rchc@maths.soton.ac.uk*
Ind. liaison chair	Richard Zobel, *r.zobel@ntworld.com*
Conf. venue chair	John Pollard, *j.pollard@ee.ucl.ac.uk*
Repr. EUROSIM	A. Orsoni, *A.Orsoni@kingston.ac.uk*
Edit. Board SNE	A. Orsoni, *A.Orsoni@kingston.ac.uk*

CEA-SMSG – Spanish Modelling and Simulation Group

CEA is the Spanish Society on Automation and Control In order to improve the efficiency and to deep into the different fields of automation, the association is divided into thematic groups, one of them is named 'Modelling and Simulation', constituting the group.

→ `www.cea-ifac.es/wwwgrupos/simulacion`

⌧ CEA-SMSG / María Jesús de la Fuente,
System Engineering and AutomaticControl department,
University of Valladolid,
Real de Burgos s/n., 47011 Valladolid, SPAIN

President	María J. la Fuente, *maria@autom.uva.es*
Repr. EUROSIM	María J. la Fuente, *maria@autom.uva.es*

LSS – Latvian Simulation Society

The Latvian Simulation Society (LSS) has been founded in 1990 as the first professional simulation organisation in the field of Modelling and simulation in the post-Soviet area. Its members represent the main simulation centres in Latvia, including both academic and industrial sectors.

→ `briedis.itl.rtu.lv/imb/`

⌧ LSS / Yuri Merkuryev, Dept. of Modelling
and Simulation Riga Technical University
Kalku street 1, Riga, LV-1658, LATVIA

President	Yuri Merkuryev, *merkur@itl.rtu.lv*
Repr. EUROSIM	Yuri Merkuryev, *merkur@itl.rtu.lv*

ROMSIM – Romanian Modelling and Simulation Society

ROMSIM has been founded in 1990 as a non-profit society, devoted to both theoretical and applied aspects of modelling and simulation of systems. ROMSIM currently has about 100 members from both Romania and Republic of Moldavia.

→ `briedis.itl.rtu.lv/imb/`

⌧ LSS / Yuri Merkuryev, Dept. of Modelling
and Simulation Riga Technical University
Kalku street 1, Riga, LV-1658, LATVIA

President	Florin Stanciulescu, *sflorin@ici.ro*
Vice president	Florin Hartescu, *flory@ici.ro*
Secretary	Zoe Radulescu, *radulescu@ici.ro*
Repr. EUROSIM	Florin Stanciulescu, *sflorin@ici.ro*
Deputy	Florin Hartescu, *flory@ici.ro*
Edit. Board SNE	Florin Stanciulescu, *sflorin@ici.ro*

Simulationists

Lukas D. Schuler

Dr. sc. nat. Lukas Schuler

xirrus GmbH
Simulation

Buchzelgstrasse 36
CH-8053 Zürich, Switzerland

Tel: +41-44-741 01 74
Fax: +41-44-741 01 73

lukas.schuler@xirrus.ch

Dr. Lukas Schuler is head of simulation modeling and user experience design at the xirrus GmbH, Zurich.

Being fascinated by the world of the largest and smallest scales, Lukas Schuler studied biochemistry at the Swiss Federal School of Technology (ETH) in Zurich, while he spent his private time on astronomical projects like his own desktop planetarium for Apple Macintosh. At the end of his master study he learned to teach chemistry and then got in contact with the groups of Prof. Dr. Luisi (experimental supramolecular studies) and Prof. Dr. van Gunsteren (computer simulation of biomolecular systems) at ETH Zurich. During his PhD studies with Wilfred van Gunsteren he got strongly involved in the parametrisation of a new Molecular Dynamics (MD) force field for aliphatic hydrocarbons to be applied to the long chain tails of lipid systems. Since then he knows the essence of parametrisation, reachable accuracy and limitations of such systems by heart. The parameters he developed are still used nowadays. His fascination of the small scales he keeps alive as an advisor for the NanoImpactNet in molecular modeling and simulation as part of an investigation methodology.

After he and his companion left ETH and founded xirrus GmbH, the main focus was set on software development of Internet based applications for a variety of customers from SMEs to government. In 2001 it was hard to imagine that one day the methods he applied to molecular systems would be a valuable approach for his customers from research and development, just because the computational resources are at hand 6 years later.

In the meanwhile he learned many tricks of usability design, relational databases, and rock solid reliable software services. This knowledge now serves as a fundament for acquisition of new simulation projects. His company has set its focus on microsimulations. Whether this is still on biomolecular systems, on nanoparticles, or on a peer to peer communication network, he likes to apply his knowledge to fresh models that his customers come up with.

Through his network he keeps contact to researchers from molecular dynamics, surface designers for nanotechnology, environmental specialists and many other disciplines.

He designed a plant dynamics simulation project to simulate the potential spread of genetically modified crops in Switzerland on behalf of the National Research Program 59 by the Swiss Science Foundation. The project involved the leading Swiss crop plant researchers and aimed at complementing the very limited field studies with large-scale geographic simulation and economic assessment.

Often, he assumes to be too visionary for the time being. In the area of warning systems he sees potential for microsimulation in two ways: one simulation should investigate the physical properties of the communication systems to be improved. The other simulation might consider the human reactions to the systems and their impacts to seek for optimal results. The two should not be considered at once, since the technical details are of much higher precision than the human behaviour can be dealt with.

He seeks to convince leading SME to keep their innovation pace at high levels by investigating their open questions that they could not answer by experiments through simulation.

He joined ASIM in 2006 to be involved in a network of simulation competence and get the possibility to publish some of his work within. In general, it is better to not only represent a private company in the market, but also show interest to research and development of the community.

Lukas D. Schuler
lukas.schuler@xirrus.ch

COMSOL MULTIPHYSICS™

Proceedings CD der Konferenz zur Multiphysik-Simulation

ANWENDUNGSBEREICHE:

- Akustik und Fluid-Struktur-Interaktion
- Brennstoffzellen
- Chemietechnologie und Biotechnologie
- COMSOL Multiphysics™ in der Lehre
- Elektromagnetische Wellen
- Geowissenschaften
- Grundlegende Analysen, Optimierung, numerische Methoden
- Halbleiter
- Mikrosystemtechnik
- Statische und quasi-statische Elektromagnetik
- Strömungssimulation
- Strukturmechanik
- Wärmetransport

Bestellen Sie hier Ihre kostenlose Proceedings CD mit Vorträgen, Präsentationen und Beispielmodellen zur Multiphysik-Simulation:

COMSOL MULTIPHYSICS CONFERENCE 2005
PROCEEDINGS AND USER PRESENTATIONS CD
SPECIAL EDITION
COMSOL

TITEL, NACHNAME ______

VORNAME ______

FIRMA / UNIVERSITÄT ______

ABTEILUNG ______

ADRESSE ______

PLZ, ORT ______

TELEFON, FAX ______

EMAIL ______

Fax: +49(0)551 / 99 721- 29

www.comsol.de/conference2005/cd/

www.comsol.de

515.000.000 KM, 380.000 SIMULATIONEN
UND KEIN EINZIGER TESTFLUG.
DAS IST MODEL-BASED DESIGN.

Nachdem der Endabstieg der beiden Mars Rover unter Tausenden von atmosphärischen Bedingungen simuliert wurde, entwickelte und testete das Ingenieur-Team ein ausfallsicheres Bremsraketen-System, um eine zuverlässige Landung zu garantieren. Das Resultat – zwei erfolgreiche autonome Landungen, die exakt gemäß der Simulation erfolgten.
Mehr hierzu erfahren Sie unter:
www. mathworks.de/mbd

MATLAB® & SIMULINK®

The MathWorks

Accelerating the pace of engineering and science